Practical Electronics

The UK's premier electronics, computing and maker magazine

Circuit Surgery
Setting up and timing a DAC for DSP.

Audio Out
An audio amplifier circuit using a complementary cascode

Precision Electronics
Getting the most out of your ADC in precision applications.

Compact OLED Clock & Timer

DOWN
UP
OK
MODE
SYDNEY
2:00:52
TUE 21 MAY 2024
SILICON CHIP
COMPACT CLOCK & TIMER
WARNING! MAINS VOLTAGE ON ALL PARTS
SILICON CHIP 180VDC/230VDC Motor Controller © 2024

180-230V DC Motor Speed Controller with feedback

PLUS!

Techno Talk – humanoid robots are coming soon!
Cool Beans – generating predictably random numbers.
Staggering generator loads – using an existing design.

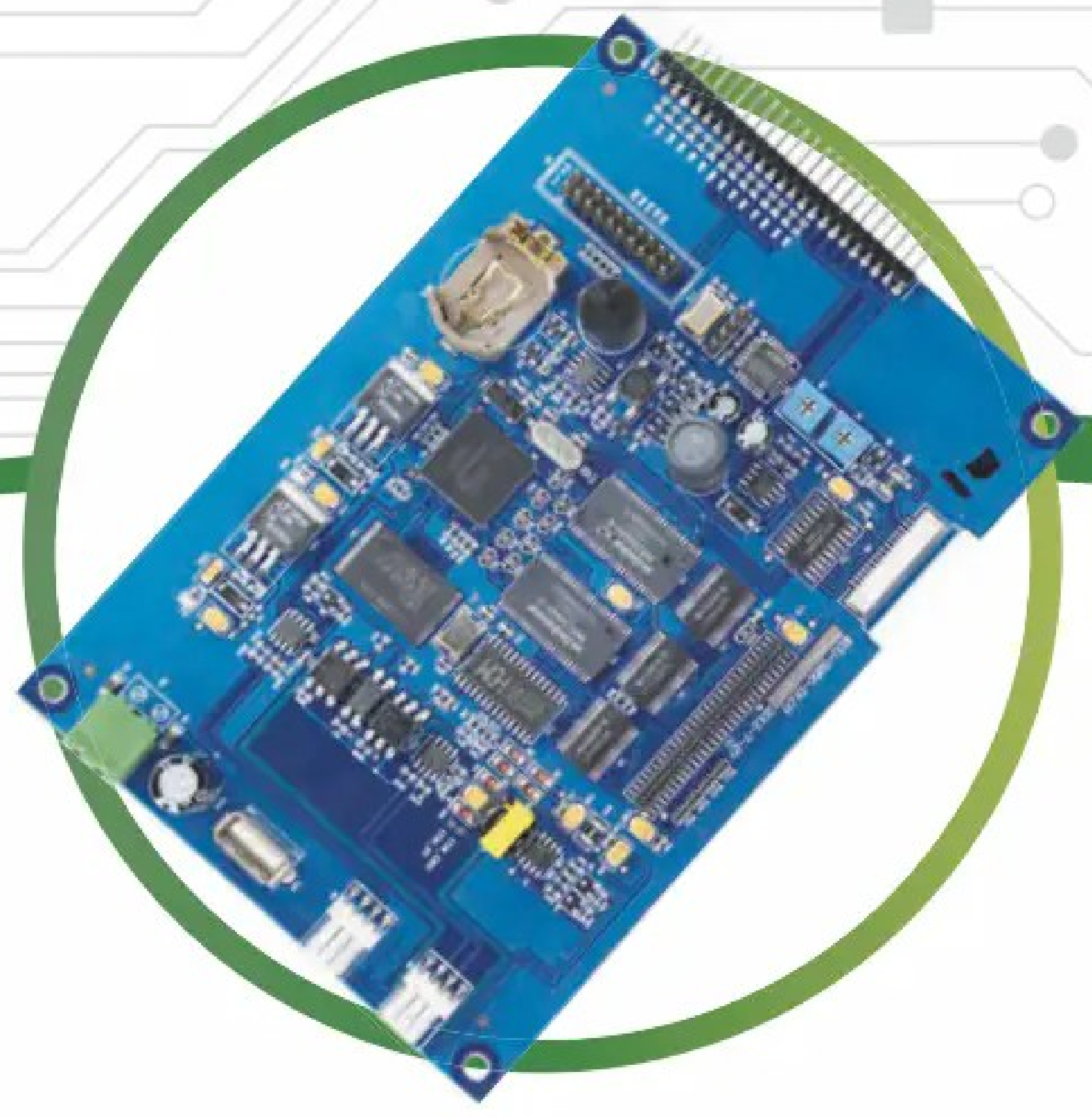
Jul 2025 £5.99

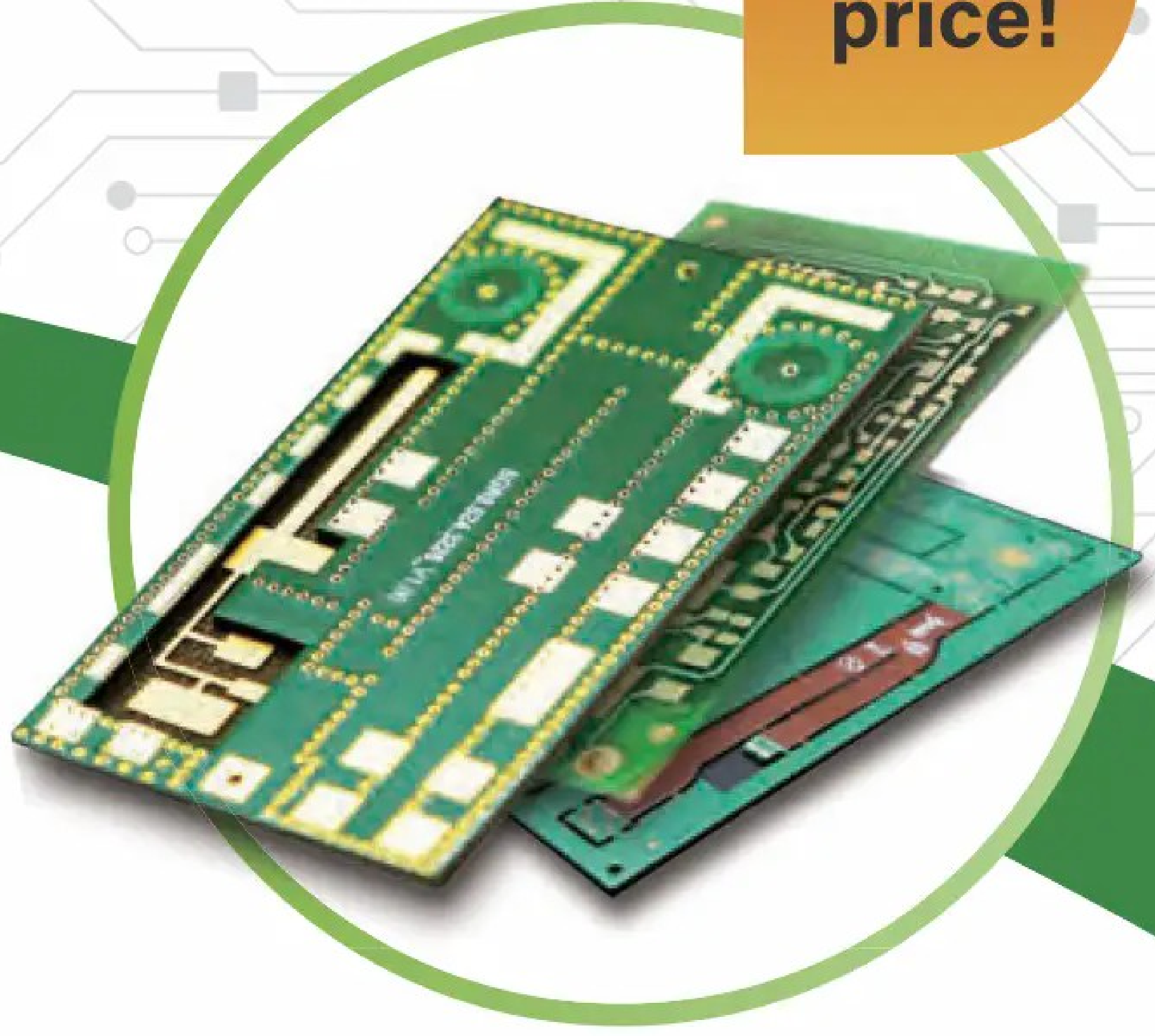
www.electronpublishing.com | @practicalelec | practicalelectronics

PCBWay

FULL-SERVICE
ELECTRONICS
MANUFACTURER

PCB Capabilities:





Special price!

PCB Assembly
From $29
- Improved assembly precision
- Short turnaround for assembly
- Flexible support

Flex, Rigid-Flex PCB
From $46.74
- Up to 16 layers Flex circuits
- Polymide Flex, PET, High Frequency
- 0.025-≥0.4 mm FPC thickness

PCB Prototype
From $5/10 pcs
- Multiple color options
- 24-hour production time
- Worldwide delivery

www.pcbway.com
service@pcbway.com

Practical Electronics

Volume 54. No. 7
July 2025
ISSN 2632 573X

Contents

DOWN
UP
OK
MODE
SLEEPING
SILICON CHIP
COMPACT CLOCK & TIMER
3945414A_Y71_231117

Made in Australia & the UK.

Written in Britain, Australia, the USA and Ireland.

Read everywhere.

© Silicon Chip Publications Pty Ltd 2025

Copyright in all drawings, photographs, articles, technical designs, software and intellectual property published in *Practical Electronics* is fully protected, and reproduction or imitation in whole or in part are expressly forbidden.

The August 2025 issue of *Practical Electronics* will be published on Thursday, 3rd of July 2025 (see p80).

Projects and Circuits

Compact OLED Clock/Timer by Tim Blythman 14
This handy, portable, rechargeable device combines a clock, timer and stopwatch and can display different time zones. It has an internal crystal and incorporates a WiFi time source, so it is always accurate, even if a leap second occurs.

180-230V DC Motor Speed Controller, part one by John Clarke 34
High-voltage DC motors are commonly used in lathes, treadmills, industrial conveyor belts and similar equipment. This Speed Controller can control such a motor over a wide range of speeds, from very slow to maximum, maintaining a constant speed.

Repurposing the Mains Power-up Sequencer by John Clarke 52
Generators and inverters are not always powerful enough to run more than one high-current appliance at a time. The Mains Power-Up Sequencer from the February & March issues can be programmed to do that automatically.

Intelligent Dual Hybrid Power Supply, part two by Phil Prosser 64
Our new Dual Hybrid Supply uses switchmode and linear regulators, so it has very quiet outputs. They can be used independently or together in series or parallel, all controlled through a single easy-to-use digital interface.

Series, Features and Columns

The Fox Report by Barry Fox 4
Robotic mowers are a labour saver, but they take some effort to set up properly.

Circuit Surgery by Ian Bell 6
Topics in digital signal processing – Implementing DSP on a microcontroller, part two.

Techno Talk by Max The Magnificent 24
HITL used to stand for hardware-in-the-loop. Now, HITL also stands for human-in-the-loop, where human judgement is added to automated systems.

Max's Cool Beans by Max The Magnificent 26
Weird & Wonderful Arduino Projects, part 7: predictably random.

Precision Electronics, part seven by Andrew Levido 44
Last month, we covered the various sources of analog-to-digital and digital-to-analog conversion errors. This month, we will focus on analog-to-digital converters (ADCs) and, as usual, we'll include a practical example.

Audio Out by Jake Rothman 57
Pretty much all linear amplifier circuit designs these days use the standard long-tailed pair (LTP) at their input. There's another way to do it, though, using a complementary cascode. That method has some advantages, so I decided to give it a try.

Regulars and Services

Editorial 2
ChatGPT can analyse circuit diagrams

Reader services – Editorial and Advertising Departments 2

Subscribe to Practical Electronics and save money 3

Practical Electronics Back Issue DOWNLOADS – 2024 now available! 33

Teach-in Bundle – what a bargain! 51

Practical Electronics PCB Service 76
PCBs for Practical Electronics projects

Classified ads and Advertiser index 79

Next month! – highlights of our next issue of Practical Electronics 80

Renewals paid by PayPal, Visa and Mastercard are now handled via our website: www.electronpublishing.com

Practical Electronics

Volume 54. No. 7
July 2025
ISSN 2632 573X

Editorial

Editorial offices
Electron Publishing (Australia)
Tel +61 2 9939 3295
Email pe@pemag.au
Web www.electronpublishing.com

Address mail to:
Electron Publishing (Australia)
PO Box 194, Matraville NSW 2036
Australia

Advertising enquiries
+61 2 9939 3295 pe@pemag.au

Editor Nicholas Vinen
Publisher Nicholas Vinen
Digital subscriptions Stewart Kearn **Tel** 07918 614662
Online Editor Alan Winstanley
Web Systems Kris Thain
Production Bao Smith
Technical staff Tim Blythman, John Clarke

Print subscriptions
Practical Electronics Subscriptions
PO Box 6337
Bournemouth BH1 9EH
United Kingdom
Tel 01202 087631
Email pesubs@selectps.com

Technical enquiries
We regret that technical enquiries cannot be answered over the telephone. We are unable to offer any advice on the use, purchase, repair or modification of commercial equipment or the incorporation or modification of designs published in the magazine.

Questions about articles or projects should be sent to the editor by email: pe@pemag.au

Projects and circuits
All reasonable precautions are taken to ensure that the advice and data given to readers is reliable. We cannot, however, guarantee it and we cannot accept legal responsibility for it.

Some projects and circuits published in *Practical Electronics* employ voltages that can be lethal. Do not build, test, modify or fix any mains-powered equipment unless you fully understand the safety aspects involved and you use an RCD (GFCI) adaptor.

Component supplies
Silicon Chip Publications may offer kits or other parts for making our projects, but not in all cases. When kits are not available, readers will need to find and source parts themselves. We advise readers to check that all parts are still available before commencing any project in a back-dated issue.

Advertisements
Although the proprietors and staff of *Practical Electronics* take reasonable precautions to protect the interests of readers by ensuring as far as practicable that advertisements are bona fide, the magazine and its publishers cannot give any undertakings in respect of statements or claims made by advertisers, whether these advertisements are printed as part of the magazine, or in inserts. The Publishers regret that under no circumstances will the magazine accept liability for non-receipt of goods ordered, or for late delivery, or for faults in manufacture.

Transmitters/bugs/telephone equipment
We advise readers that certain items of radio transmitting and telephone equipment which may be advertised in our pages cannot be legally used in the UK. Readers should check the law before buying any transmitting or telephone equipment, as a fine, confiscation of equipment and/or imprisonment can result from illegal use or ownership. The laws vary from country to country; readers should check local laws.

ChatGPT can analyse circuit diagrams

I took a circuit I had drawn in CorelDraw and removed anything obvious regarding how it functioned (it's reproduced below). I then fed the image to ChatGPT and asked it what the circuit did.

I thought it would get it wrong as it requires quite a few logical inferences to figure out but it described its function almost perfectly! Its only mistake was with the function of the diodes; they're necessary because the two outputs drive some of the same LEDs. Here's an abbreviated version of what it said:

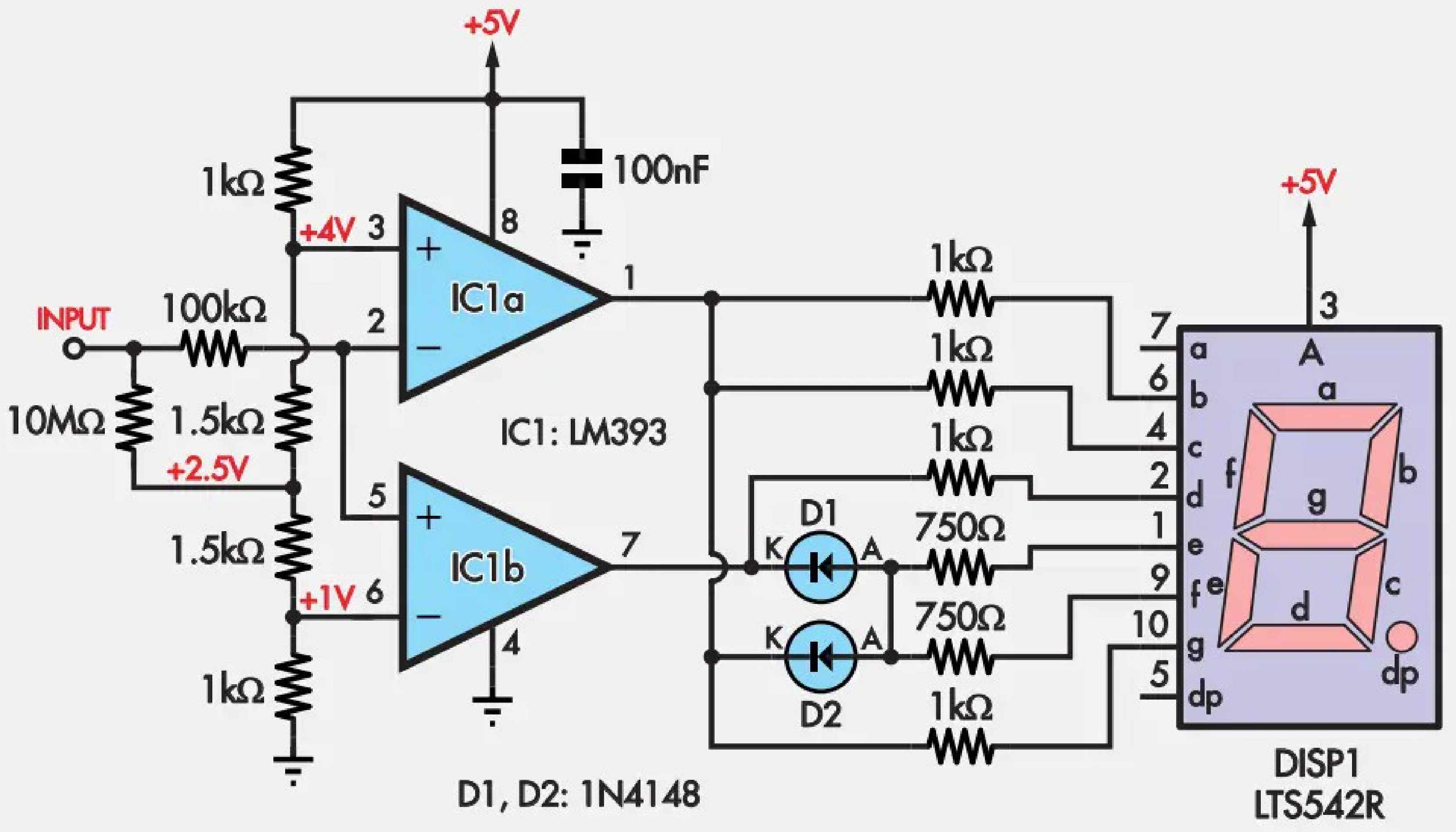
This is a high/low voltage indicator circuit using an LM393 dual comparator and a 7-segment display (LTS542R) to visually indicate the input logic level — high or low.

IC1a and IC1b are configured as voltage comparators. Each comparator compares the input to a reference level (likely derived from resistor networks and diode drops). When the input crosses a threshold, the corresponding comparator output changes.

The outputs of the comparators drive specific segments of the 7-segment display to show either "H" (High) or "L" (Low). The series resistors limit the current through each segment. The arrangement of resistors and wiring determines which segments light up based on which comparator is active.

Diodes D1 & D2 provide voltage drops to slightly offset the comparator thresholds — possibly giving hysteresis or creating slightly different high/low switching points.

If input is LOW (0V), one comparator activates and the display shows "L". If the input is HIGH (5V), the other comparator activates and the display shows "H".



Nicholas Vinen, Electron Publishing (Australia)*
Publisher & Editor,
Practical Electronics Magazine

* a division of Silicon Chip Publications Pty Ltd.

Practical Electronics

UK readers SAVE £1 on every issue

SUBSCRIBE NOW!

01202 087631

Take out a one-year subscription and save more than £10 over the year.
Even better – save over £1 per issue if you subscribe for two years – a total saving of £26.

Overseas rates represent exceptional value

You also:
- **Avoid any cover price increase for the duration of your subscription**
- **Get your magazine delivered to your door each month**
- **Guarantee a copy, even if your local newsagent sells out**

Order by:
- **Phone or post with a cheque, postal order or credit card**
- **Or order online at: www.electronpublishing.com**

Practical Electronics

SUBSCRIPTION PRICES

Subscriptions for delivery direct to any address in:

UK: 6-months £33.99, 12-months £59.99, 24-months £114.99

Europe Airmail: 6-months £38.99, 12-months £69.99, 24-months £129.99

Rest Of The World Airmail: 6-months £46.99, 12-months £84.99, 24-months £164.99

Cheques (in **£ sterling only**) payable to ***Select Publisher Services Ltd*** and sent to: Practical Electronics Subscriptions, PO Box 6337, Bournemouth BH1 9EH, United Kingdom

Tel: 01202 087631
Email: pesubs@selectps.com

Also via our online shop at**: www.electronpublishing.com**

Subscriptions start with the next available issue. We accept MasterCard or Visa.
(For past issues see the Back Issues page.)

DIGITAL SUBSCRIPTIONS

Electronic subscriptions are available from £29.99 for 12 months, more details at: **www.electronpublishing.com**

SUBSCRIPTION ORDER FORM

☐ **6 Months:** UK £33.99, Europe £38.99 (Airmail), Rest of the World £46.99 (Airmail)
☐ **1 Year:** UK £59.99, Europe £69.99 (Airmail), Rest of the World £84.99 (Airmail)
☐ **2 Years:** UK £114.99, Europe £129.99 (Airmail), Rest of the World £164.99 (Airmail)

To: **Practical Electronics Subscriptions**
PO Box 6337
Bournemouth BH1 9EH
United Kingdom

Tel: 01202 087631
Email: pesubs@selectps.com

I enclose payment of £ ....
(cheque/PO in £ sterling only)
payable to ***Select Publisher Services Ltd***

☐ Please charge my Visa/Mastercard

My card number is: ....
Please print clearly, and check that you have the number correct

Card Security Code .... Valid From Date.......................
(The last 3 digits on or just under the signature strip)

Card Ex. Date ....

Name ....

Address ....

Post code .... Tel. ....

Email ....

Subscriptions start with the next available issue.

The **Fox** Report

Barry Fox's technology column

Robotic mowers take some effort to set up properly

It sounds so appealing – instead of walking round a lawn, pushing a manual mower, or trying to keep up with a wayward petrol engine, or paying someone else to do it for you – just buy one of the many new robot lawn mowers now flooding the market.

If only it were that simple. I ran a practical test, buying the PAMR 500A1 made by Parkside and sold by Lidl for the too-good-to-miss price of £199. That's half the cost of one of the favourite alternatives, the Lawnmaster L10. Although all these devices have sensors to detect big obstacles, both the PAMR and L10 rely for guidance on signals radiated by a boundary wire laid around the lawn.

For the Lidl mower, the wire guides the mower clockwise (by ± polarity) around the lawn when its rechargeable Li-Ion battery is running flat. In theory, the mower self-docks with the charger, but getting the wire in just the right position for an easy path 'home' needs careful planning.

Again, in theory, the wire can be pegged to the turf surface, and this is a good way to find out where the device gets stuck trying to navigate tight corners or gets too close to the lawn plant borders. But as I soon found out the hard way, even when the wire is tightly pegged down, the mower is liable to cut the wire when set for cutting short grass.

The only long-term fix is to bury the wire a couple of inches. Even if the turf is sliced to make a flap, rather than dug to create a trench, it's a tedious, tough job. Sections of the boundary wire can run close together; for example, to skirt obstacles. Still, they must not cross over each other.

By now, you will be getting the picture. Anyone who buys a robot mower with the happy hope that they can unbox, plug in, pour a drink and sit in the sun watching their lawn being beautified is in for a rude awakening.

In the case of the Parkside/Lidl beast, things are much worse because – although the mower is surprisingly sturdy – the accompanying printed and online instructions are some of the worst I have ever encountered.

Packed in the box are three printed manuals, one for the battery (which is screw-sealed inside the mower casing) and two similar but not identical 40ish-page booklets. There's also a colour picture sheet without any text, just green ticks and red crosses, plus a QR code for decidedly iffy online help.

The online help is in German, so I had to auto-translate it to English. It links to an online manual, which I hoped to search by keyword for an explanation of mysterious bleeping during normal operation, and also how to fit a metal guide rail included in the box. Entering the mower model number throws up the error message "Unfortunately we couldn't find anything".

Online Help also offers a video on "Laying Boundary Cables". But clicking the play link gave the message "This video is private – if the owner has granted you access, please sign in". The same thing happened with a video on "First Mowing Pass".

The Parkside PAMR 500A1 on its charging dock.

I was then diverted to an opportunist advert for a "Secret New Invention" which is a "new antenna changing the way people watch TV and cable providers can't figure out what to do" from getwavetune.com

The instructions (https://parkside-diy.com/de/garten/rasenmaeher-vertikutierer/parkside-maehroboter-pmrda-20-li-b2-smart) refer to a phone App and Bluetooth control, but this applies to different mower models, which Lidl is not selling.

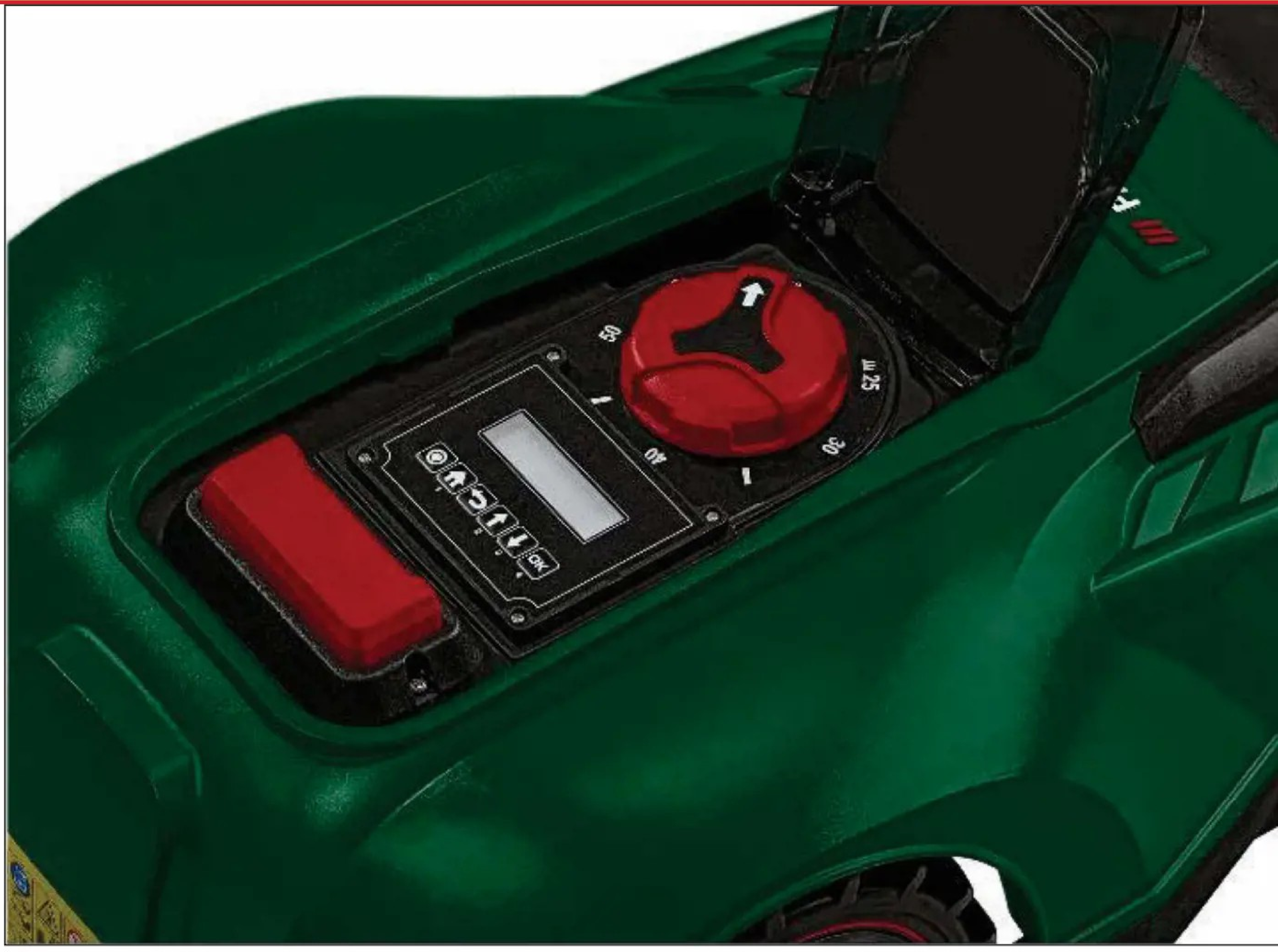
The whole experience is like assembling flat-packed furniture, but also requiring computer skills, strength for physical digging, and cleverness at second-guessing what's not clear or not visible. It is a real shame, because if and when the user gets the mower working, it works well and delivers a tidily cut lawn.

What users need is a simple Getting Started guide with basic essentials clearly stated, before getting bogged down in a welter of detail about special features, such as programming the mower to start and stop at pre-set times and cut measured zones.

So I have put together the kind of idiot's guide that Lidl so signally fails to provide. If these devices end up near-new and cheap on eBay or in second-hand shops, because their owners have never got them working, these simple basics could save a lot of people a lot of grief and money.

On first use, it's obligatory to enter the date and time using the on-board buttons and LCD strip screen. This is fairly intuitive. Press the On/Off button for a few seconds, use the Up/Down arrow keys to set year, month, day and time of day. Press OK after each entry.

It's also necessary to set a PIN for 'unlocking' the mower every time is it is used. Press OK and use the numbers 1–4 shown on four of the buttons. Repeat the entry to confirm the new PIN and press OK. But beware; I realised only just in time that I was setting a



The control panel and cutting height adjustment are under a waterproof lid.

code without knowing what numbers the displayed asterisks denoted.

This would lock the user out, with no obvious way to over-ride and reset. I didn't dare experiment, so I just chose 1234 – which rather defeats the object of setting a PIN.

It's good practice to leave service loops, for later adjustment, when connecting the charger leads and boundary wires. But the loops must be tucked away behind the charger, out of mower pickup range.

Be sure to obey the 30/35 cm boundary rule; leave at least 30cm, and ideally 35cm, from the lawn border. For who knows what design reason, the mower has to pass well over the wire before it can detect the guide signal and performs a multi-point turn. [It's probably because the sensor has to be mounted centrally in the mower so that it doesn't matter what angle it approaches the wire – Editor]

Other than backing away from the boundary wire (which carries 21.5V DC, modulated at between 100kHz and 148.5kHz), and following the wire (negative to positive) when returning to the charger, the mower moves entirely at random.

There are no straight mow lines. The cut grass is spread back on the lawn and mulches itself out of sight. This works surprisingly well. After a few days of random cutting, the lawn looks good.

The device needs a 100% charge before it can be operated while still docked, but if pulled out of the dock, it obeys commands when only 30% charged. It beeps when it hits an immovable object, but also beeps every so often while happily mowing; that is probably to signal its battery status.

The battery that comes with the mower is one of the 20V units that are used by some (but not all) of Lidl's other Parkside tools. Its capacity is 2Ah, but I unscrewed the battery housing and did a plug-swap with a 4Ah battery.

A rain sensor stops everything when the weather is wet. As long as the weather is dry, and the mower is charged, the basic essential commands are as follows:

Press On/Off button to get a 'Welcome' message and PIN request. Enter your pre-set PIN to get the message 'Standby.'

From 'Standby', a short press of the On/Off button gives the message "Start Manual". After that, press the OK button to get the message "Close Cover to Start". Close the cover (which physically protects the buttons and LCD) to trigger multiple happy beeps before the mower starts mowing.

To stop mowing, lift the cover or press the Big Red Emergency Stop button. The mower can then be re-started only by returning to the Welcome/Standby procedure.

In theory, the mower takes itself back to the charging station and docks itself when the battery gets hungry with 30% charge; in practice, this only works if the ground is even and the boundary wire is at just the right position in relation to the dock. Trial and error are needed. This is when service loops are needed.

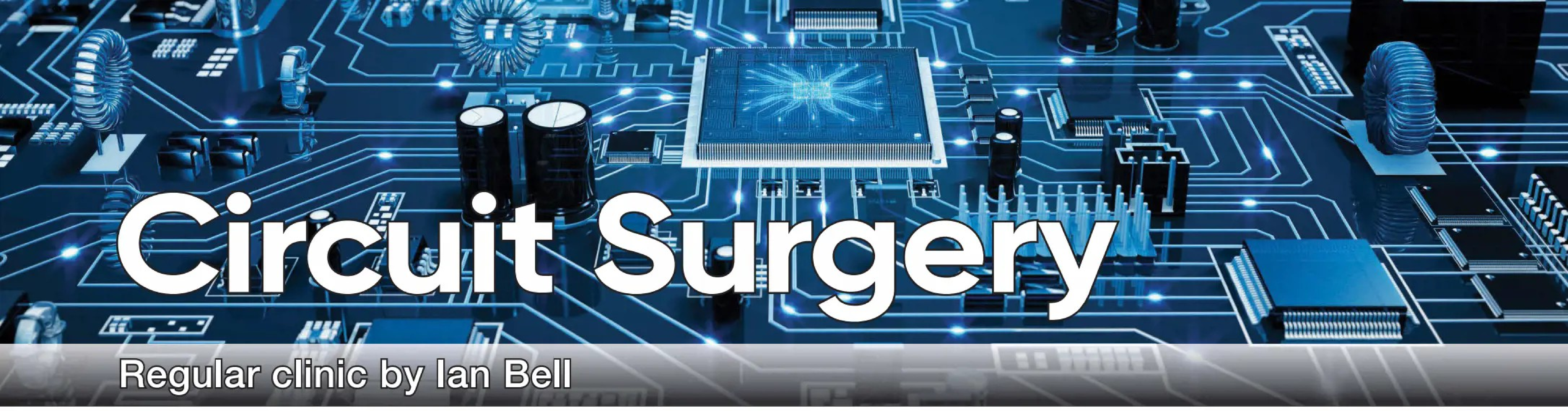
The mower can be manually sent back to self-dock by a modified Start procedure. From Standby, long press Home and close the cover to get happy bleeps and start. Note that when returning to the charging station, the mower stops mowing; its blade wheel does not turn.

The operating time default is eight hours per day with Rain sensing On. This and other tweak functions, like setting distance (dst)% and ratio percent (pct)% spot areas, can be altered by starting the PIN Start procedure, then using the Up-arrow button to display Settings menus and sub-menus controlled by the arrows and back buttons.

I suspect that most people, like me, will just be thrilled to have finally got some tangible reward for their £200 and quit while ahead! PE

£249
£199
24/04/25 - 30/04/25‡
Lidl
Plus

It was tempting...



Circuit Surgery

Regular clinic by Ian Bell

Topics in digital signal processing – Implementing DSP on a microcontroller, part two

We are looking at various topics related to digital signal processing (DSP).

DSP covers a wide range of electronics applications where signals are manipulated, analysed, generated, stored or displayed as digital data but originate from and/or are converted to real-world signals for interaction with humans or other parts of the physical world.

Fig.1 shows the key elements of a generic DSP system with a signal path from an analog input via digital processing to an analog output.

This does not necessarily represent every DSP system (not all have all the parts shown), but it serves as a reference for the various subsystems we will look at.

Last month, we introduced an example microcontroller-based DSP implementation that we will be working through. The aim is to implement a sinc filter, which we discussed in detail in earlier articles.

We will use a DSP-capable processor from the STM32 family from ST Microelectronics on a B-L475E-IOT01A development board. However, other suitable processors and boards from the STM32 series could also be used.

Last month, we gave an overview of some relevant aspects of the development board and ST's development software (STM32CubeIDE), relating this to the generic DSP structure in Fig.1.

We will not give a full step-by-step walk-through on using the software. Therefore, if you would like to try these examples and are not already familiar with STM32CubeIDE, you should look at some online tutorials that cover basic projects, such as LED blinkers, to gain familiarity with the development tools.

STM32CubeIDE can automatically generate the code required to initialise the microcontroller, configure pin usage and set-up on-chip peripheral hardware, such as the digital-to-analog converter (DAC). The requirements for a given project are set via the Device Configuration Tool.

On saving the configuration, the software generates or updates the main code file for the project, writing the code required to set up the processor for the chosen development board and any user options from the Configuration Tool.

The software installation includes a library of functions, referred to as the Hardware Abstraction Layer (HAL). This makes it easy to control the microcontroller and its peripherals without having detailed low-level knowledge of its hardware, such as knowing what binary codes to write to specific registers to achieve a hardware operation.

These are used in the automatically generated configuration code, and should also be used where appropriate in user-written code (eg, to write a value to the DAC).

Before looking at the implementation of filters, we will discuss individual aspects of the system: the DAC and ADC (analog-to-digital converter), sample timing and data transfer. We will look at simple approaches and their disadvantages and more advanced approaches that make better use of the microcontroller's resources.

We will provide code examples to illustrate the principles discussed. These can be run as fully operational projects on the development board.

First we will consider controlling the DAC, which leads naturally to how sample timing can be organised.

Initial DAC setup

To test the basic operation of the DAC, we can make it generate a simple waveform and view it on an oscilloscope. Our final goal is to implement a filter, not a waveform synthesis system, so we will keep this very straightforward and just generate a stepped sawtooth (see Fig.2). This is simple to code, and includes some large jumps in output voltage, which will show how fast the DAC output can change.

The step waveform is generated by initially setting the DAC output to *DAC_Start*, then subtracting *DAC_Step* each time a new value is required until the current value written to the DAC (*DAC_Value*) is less than the step value. At this point, the value will be set back to *DAC_Start*, and the process is repeated.

We will use relatively large steps; stepping the DAC through every code value would result in a long, slow slope that may be difficult to resolve to individual steps on a basic oscilloscope.

The waveform in Fig.2 is shown as a positive-only voltage, as this is what is available on the development board. The DAC output voltage range is determined by its reference voltage inputs. On the B-L475E-IOT01A1 board the VREF+ pin is connected to the 3.3V analog power supply (VDDA) and the VREF- pin is connected to ground (0V), as per page 3 of the schematic (**https://pemag.au/link/ac5j**).

The DAC output voltage therefore ranges from 0V to 3.3V. The DAC is a 12-bit device, so the maximum for *DAC_Value*

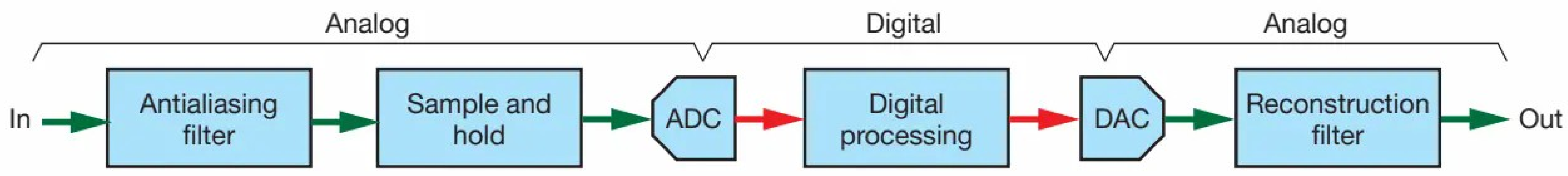


Fig.1: a generic digital signal processing (DSP) system structure.

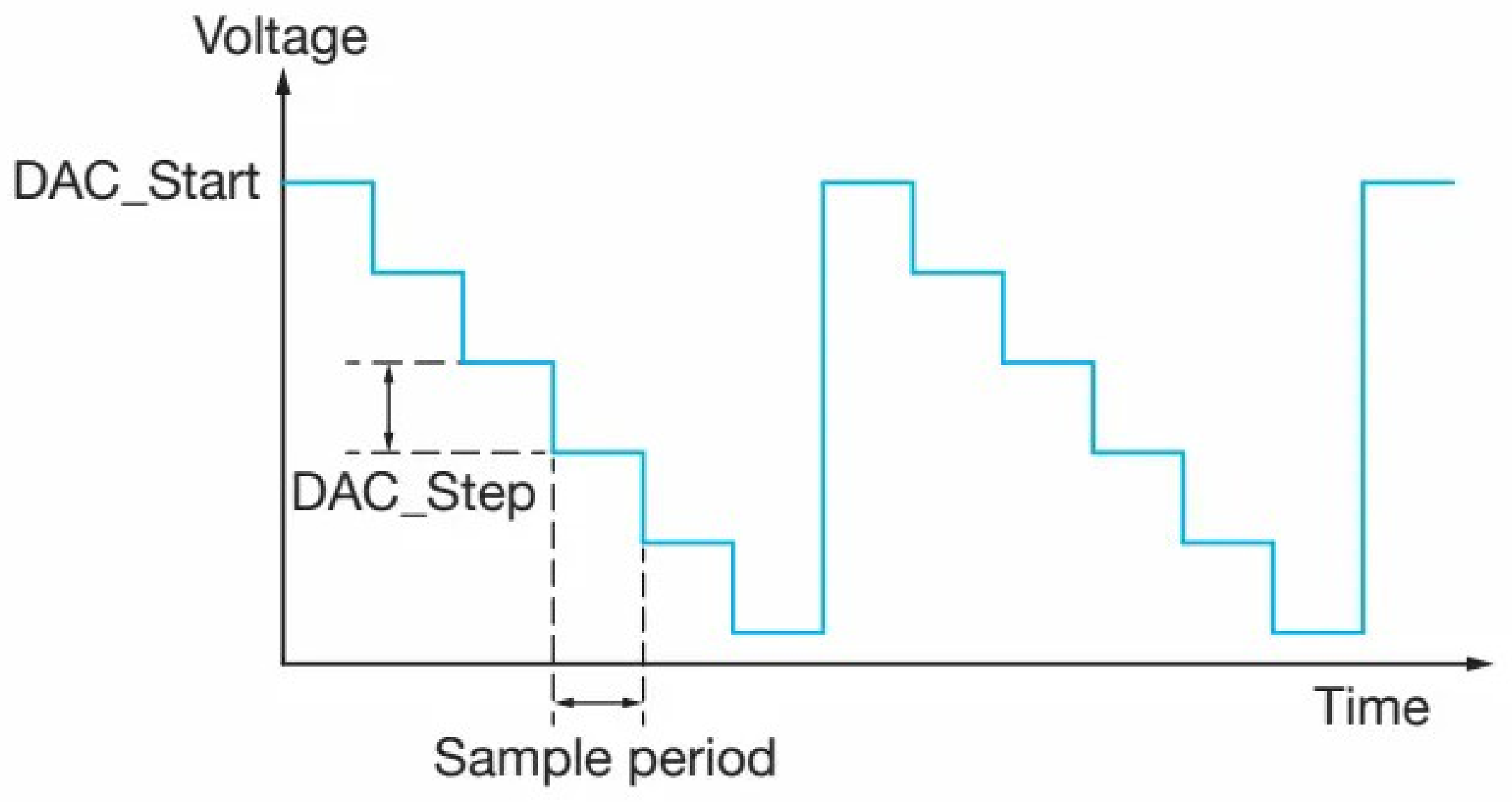


Fig.2: the desired test signal from our DAC.

is $2^{12} - 1 = 4095$, and the output voltage is given by:

$$DAC\_OUT = VREF+ \times DAC\_Value \div 4095$$

So, in this case:

$$DAC\_OUT = 3.3 \times DAC\_Value \div 4095$$

It is interesting to see how fast the DAC can be updated, so initially, no attempt will be made to set the sample period to a specific value. The code will comprise a free-running loop to calculate and apply the waveform described above.

It is useful to have a digital output to indicate when a new value was sent to the DAC. Bringing a digital output high before a function is called to update the DAC and bringing it low again afterwards will result in waveforms as shown in Fig.3. The time between corresponding edges of the pulse waveform in successive cycles will be equal to the sampling time.

The code required to create the waveforms in Figs.2 & 3 can be defined by the pseudocode shown in Listing 1.

Auto-generated and user code

Last month, we looked at the microcontroller's DAC hardware and development board schematic, and determined that we will use DAC1's output channel 2 on pin PA5. To briefly recap, this requires changing the pin usage of PA5 to DAC1_OUT2 in the Device Configuration Tool.

We also need to configure the DAC itself (DAC1 in the Analog section of the Categories list). Here, in the Mode panel we change "OUT2 connected to" from "Disable" to "only to external pin" – this will enable the DAC and route its output signal to the pin. After making these changes, save the configuration and that will generate C code for the project.

The generated **`main.c`** file contains about 700 lines of code, not all of which is directly relevant to this project. A lot of it configures the processor to be compatible with the development board. As is typical for embedded software, there is initialisation/configuration code that runs once, upon board reset or power-up, after which the program enters an infinite loop where the main operations are performed.

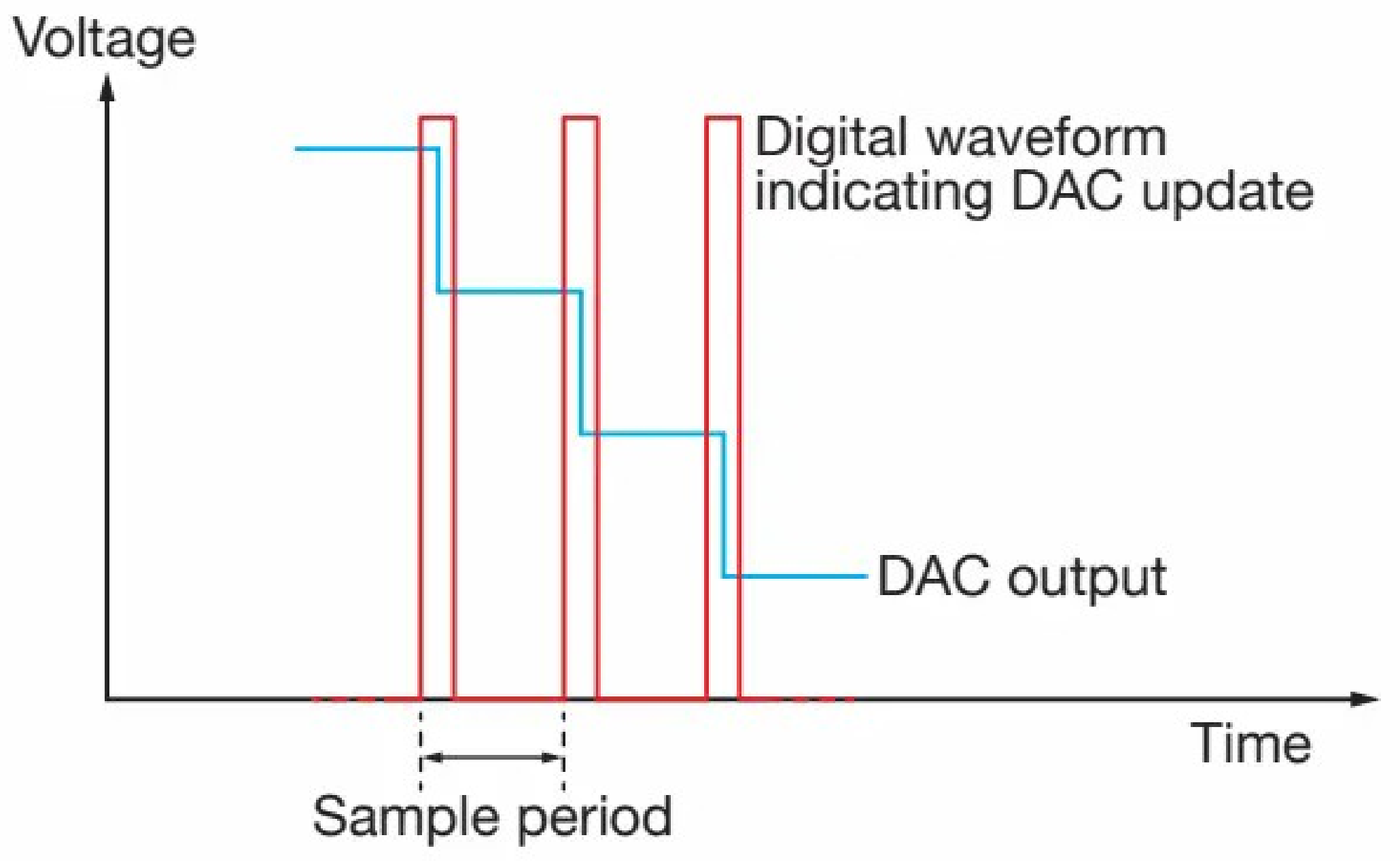


Fig.3: pulses from an output help us measure the DAC output's update timing.

The generated main loop is initially empty, so the program will initialise the hardware and then enter an infinite loop that does nothing.

We need to add code before the main loop to define any variables and such that we require, plus code to perform any initialisation that is not automatically generated. We also need to write the code for the main loop. We can add functions to the file and call these from our code.

As well as the main loop, we can also write code that is triggered to run by events in the hardware – we will be discussing this in more detail later.

Listing 2 shows the code added to the auto-generated **`main.c`** file to implement the stepped DAC waveform, which is based on Listing 1. The variables discussed above are declared on lines 59 to 61. The code for the main loop is on lines 127 to 142.

All of our code is placed between corresponding "User Code" start and end comments. We must always do this to prevent the code we add from being overwritten by any configuration updates. Some user code sections are expected to be used for specific purposes; for example, "PV" (Listing 2, line 58) is for private variables. Others are more general and just numbered.

The main loop has two user sections, making it relatively easy to mistakenly add code between the sections (Listing 2, line 130), which would be erased after making any configuration changes. So avoid doing that.

```
Initialize DAC
Start DAC converting
DAC_Value = DAC_Start
while (1)
{
  if (DAC_Value < DAC_Step)
    DAC_Value = DAC_Start
  else
    DAC_Value = DAC_Value - DAC_Step

  Set digital output to 1
  Set DAC output to DAC_Value
  Set digital output to 0
}
```

Listing 1: pseudocode for generating a stepped sawtooth waveform.

Starting the DAC

Before the DAC can be used, it must be initialised to operate as needed by the design. The registers that control its operation, and the hardware on the chip that controls signal routing, must be set up as required. Most of this work is done by the code that is automatically generated by the Device Configuration Tool.

The generated **`main.c`** file contains the definition of a global variable called **`hdac1`**, ie:

`DAC_HandleTypeDef hdac1;`

This variable is used to reference the DAC whenever we perform an operation on it using the HAL code library functions.

A function named **`MX_DAC1_Init`** is called during the initialisation part of the microcontroller software, along

```
58  /* USER CODE BEGIN PV */
59  uint32_t DAC_Start = 4095; // Value at peak of sawtooth wave
60  uint32_t DAC_Value = 4095; // Current DAC output value
61  uint32_t DAC_Step = 255;   // Change in DAC value at each step (sample)
62  /* USER CODE END PV */
…
121  /* USER CODE BEGIN 2 */
122  HAL_DAC_Start(&hdac1, DAC_CHANNEL_2);
123  /* USER CODE END 2 */
124
125  /* Infinite loop */
126  /* USER CODE BEGIN WHILE */
127  while (1)
128  {
129   /* USER CODE END WHILE */
130
131   /* USER CODE BEGIN 3 */
132   // Calculate next sawtooth waveform value for DAC
133   if (DAC_Value < DAC_Step)
134     DAC_Value = DAC_Start;
135   else
136     DAC_Value = DAC_Value - DAC_Step;
137
138   // Update the DAC and pulse the D8 pin to help measure timing
139   HAL_GPIO_WritePin(GPIOB, ARD_D8_Pin, GPIO_PIN_SET);
140   HAL_DAC_SetValue (&hdac1, DAC_CHANNEL_2, DAC_ALIGN_12B_R, DAC_Value);
141   HAL_GPIO_WritePin(GPIOB, ARD_D8_Pin, GPIO_PIN_RESET);
142  }
143  /* USER CODE END 3 */
```

Listing 2: user code for controlling the DAC.

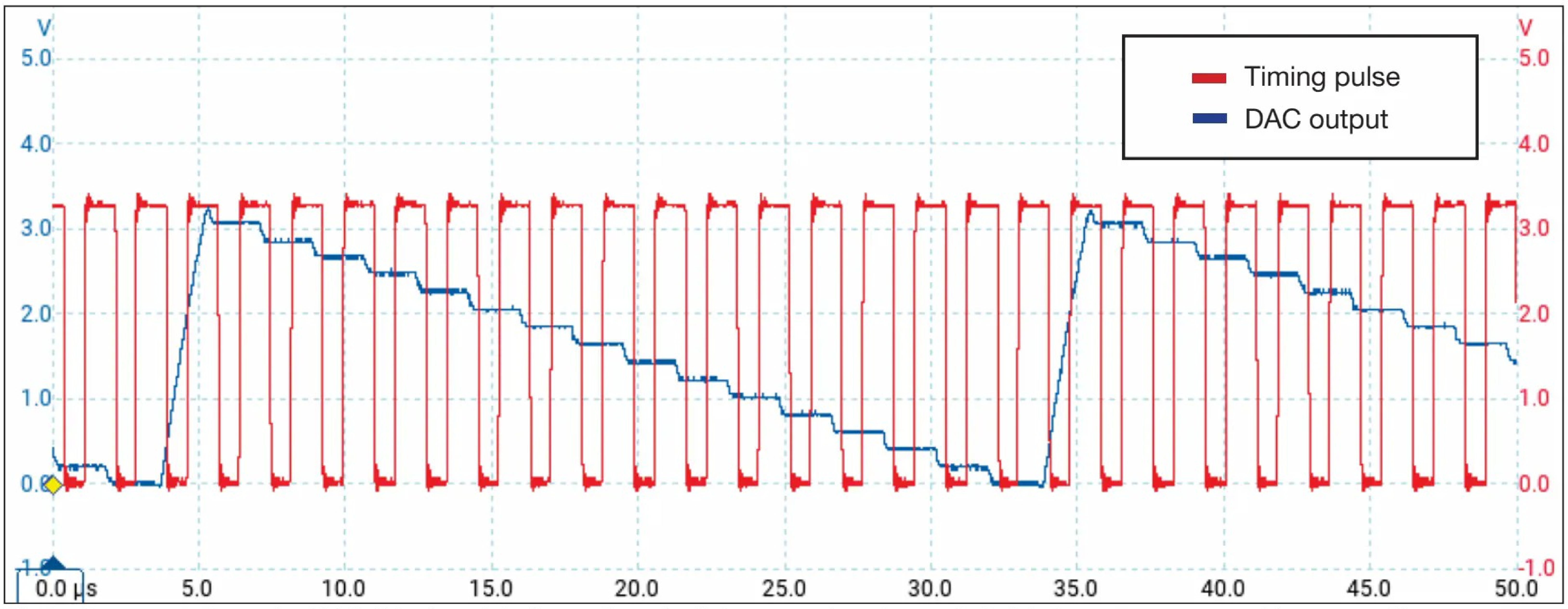


Fig.4: the output from the DAC (blue) and timing pulses (red) from running the code in Listing 2, as captured by an oscilloscope.

with various other peripheral configuration functions. It assigns the **hdac1** handle, initialises the DAC and configures output channel 2 in accordance with our configuration settings. The function code itself can be found about around a third of the way through the generated **main.c** file.

While the auto-generated code initialises the DAC and its output channel, we still must start it operating (performing conversions) by calling the **HAL_DAC_Start** function. This function needs to be added to the code before the main while() loop (Listing 2, line 122).

HAL_DAC_Start requires two parameters. The first is the handle mentioned above, while the second specifies the DAC channel to use. We're using channel 2, identified by **DAC_CHANNEL_2**, a value that is #defined in the HAL library.

It is possible to do all this from scratch by finding details of the relevant definitions, functions and how to use them in the *Description of STM32L4/L4+ HAL and low-layer drivers* user manual (**https://pemag.au/link/ac5h**).

Controlling the DAC output

To set the output voltage of the DAC, we use the HAL library function **HAL_DAC_SetValue** (Listing 2, line 140). This function has four parameters. The first two are the DAC handle address and channel, as for the start function. The third is a data alignment specifier, while the fourth is the DAC output value.

The DAC is a 12-bit bit type, but the value parameter is 32-bits, so the alignment specifier determines which of the 32 bits (eg, leftmost 12 or rightmost 12) are the DAC value.

The code in Listing 2 uses the rightmost (lower) 12 bits, as this means the DAC value is equal to the numerical value of the parameter. However, care must be taken to keep the parameter in the valid range of 0 to 4095.

The value **DAC_ALIGN_12B_R** is defined in the library to set right 12-bit alignment.

The digital output is set using the **HAL_GPIO_WritePin** function, which takes three parameters. The first identifies the GPIO port of the pin to be controlled (Port A to Port H); **GPIOA** to **GPIOH** are #defined to reference these ports. In this case, port B is being used (the pin used is PB2, ie, Port B, pin 2).

The next parameter is the pin number. The file **main.h**, which is also automatically created as part of the project, has #defines for the board signals, relating them to the pins. So the value **ARD_D8_Pin** can be used here. It would also be possible to use **GPIO_PIN_2**, as that is #defined in the HAL library for pin 2.

The final parameter is the state to set the pin to; **GPIO_PIN_RESET** and **GPIO_PIN_SET** are #defined for logic low and high, respectively.

Execution results

I captured the DAC output from running the code in Listing 2 on an oscilloscope, as shown in Fig.4; it clearly corresponds with Figs.2 & 3. The timing pulse has a frequency of 564kHz (a 1.77µs period), as measured by the oscilloscope. Remember that we are not setting or controlling this sampling rate – the DAC update is free-running.

The waveform is not quite as it should be, because the DAC takes longer than one sample period to change its output for the largest step; Fig.5 shows this in detail. The largest step is labelled as step 1, with others following from this. The positive edge of the timing pulse indicates the approximate time at which a new value is written to the DAC to request a new output value.

For the large step (step 1), the request for the next value (step 2) occurs before the output has finished changing. Thus, the DAC output does not quite reach the final value for step 1 before it starts changing to the step 2 value. Step 2

```
58   /* USER CODE BEGIN PV */
59   uint32_t DAC_Start = 4095; // Value at peak of sawtooth wave
60   uint32_t DAC_Value = 4095; // Current DAC output value
61   uint32_t DAC_Step = 255;   // Change in DAC value at each step (sample)
62   uint32_t LoopDelay = 51;   // Loop delay to set sample period
63   /* USER CODE END PV */
…
89   int main(void)
90   {
91     /* USER CODE BEGIN 1 */
92     uint32_t i;  // Loop counter for delay loop
93     /* USER CODE END 1 */
…
138    // Update the DAC and pulse the D8 pin to help measure timing
139    HAL_GPIO_WritePin(GPIOB, ARD_D8_Pin, GPIO_PIN_SET);
140    HAL_DAC_SetValue (&hdac1, DAC_CHANNEL_2, DAC_ALIGN_12B_R, DAC_Value);
141    HAL_GPIO_WritePin(GPIOB, ARD_D8_Pin, GPIO_PIN_RESET);
142
143    // Add delay
144    for (i=0; i<=LoopDelay; i++){}
145
146    }
147    /* USER CODE END 3 */
```

Listing 3: adding a delay to the DAC loop.

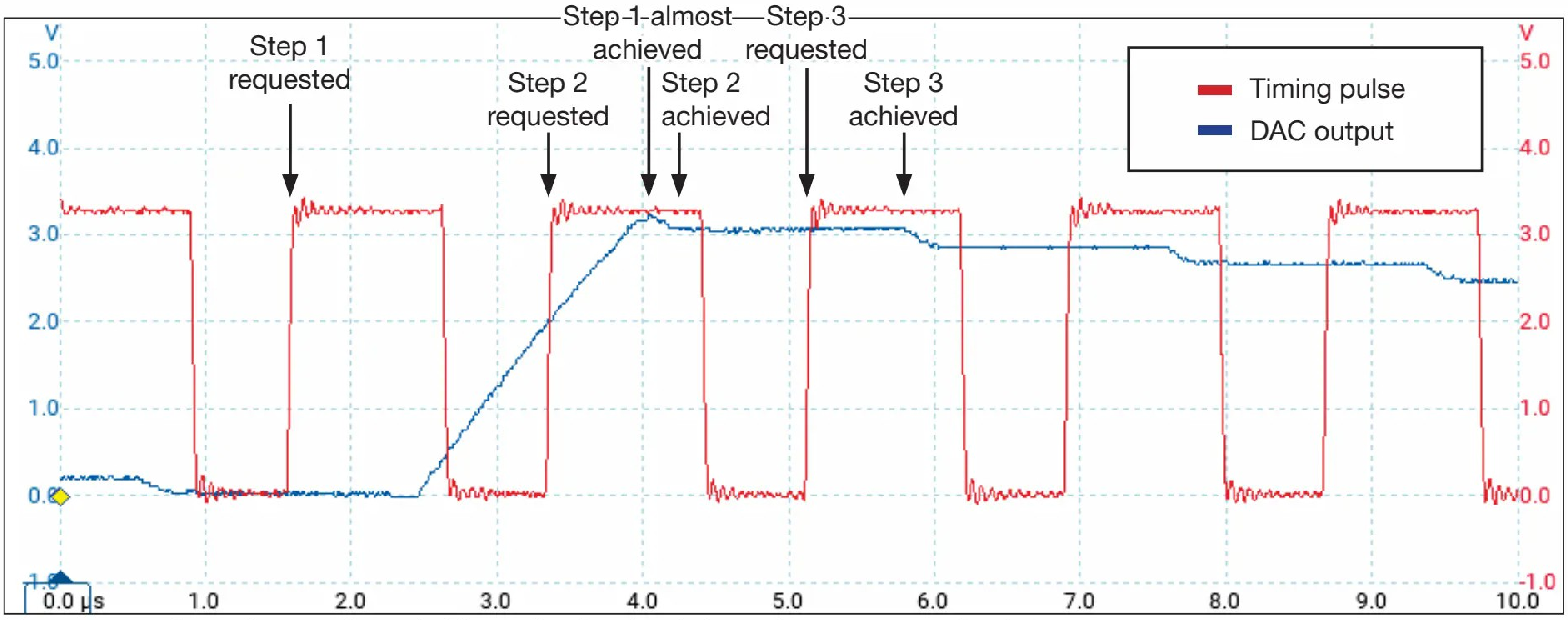


Fig.5: a zoom-in on the waveforms in Fig.4, allowing the timing to be seen more clearly.

to step 3 operates correctly due to the much smaller output voltage change.

We have no reason to expect that letting this DAC run freely would give perfect results, but this serves to show that we need to take account of the DAC's capabilities.

Setting the sampling period

The microcontroller datasheet indicates that the DAC could take up to 3µs to change its output, so a 1.77µs sample period is too short. We could slow the DAC update down by adding a delay to the main loop.

The **HAL_Delay()** function is potentially useful here; inserting a call to this in the loop could be used to control the duration of each iteration. However, **HAL_Delay()** only provides delays in milliseconds – so it can only be used to control sampling at 1ms intervals (a 1kHz sampling frequency or lower), which is far too slow for many applications.

Another problem we have is that the **HAL_Delay()** function is not particularly accurate with respect to the passed delay parameter, particularly with smaller delays. A call of **HAL_Delay(D)** produces a delay between **D** and **D+1** milliseconds.

This is because it is implemented using a free-running hardware counter, and the +1 effect is due to ensuring the delay is at least **D**, even if the function is called just before the next counter update occurs.

If the **HAL_Delay()** function is called repeatedly with a short gap between calls (as would happen in a loop with little other code), the calls will occur just after a counter update, so the delay will be close to **D+1** each time.

We can add shorter delays using a for-loop that counts for a specified number of iterations and adjust this to control the timing (see Listing 3, line 144). To use the loop, we need to declare a loop variable (Listing 3, line 92) and set a value for the number of iterations to use (Listing 3, line 62).

Fig.6 shows the output of the DAC with the for-loop delay added. The slower sampling rate means that the DAC is able to reach the required output level well before the next value is requested, even for the largest step. By experiment, a value of 51 for **LoopDelay** resulted in a sampling frequency of 96.4kHz, the closest to 96kHz (a commonly used sample frequency for audio) that could be obtained.

This illustrates a problem with this approach – it is difficult to precisely set the sample frequency. **LoopDelay** values of 50 and 52 gave 97.9kHz and 94.9kHz, respectively, so 51 was the best option, but it is not really good enough.

Another potential problem is that if the other code in the loop includes conditional operations (eg, **if… then… else…**), the number of instructions encountered on each loop iteration could be different, resulting in a varying delay value. This is bad news if we are implementing DSP that relies on a fixed sampling rate.

In conclusion, adding a delay to a loop like this is a crude technique that does not allow precise control of timing, although it is easy to do.

Hardware timers

Accurate time measurement can be readily achieved using hardware timers. These are digital counter circuits on the microcontroller that are clocked using specific, accurately controlled frequencies. Such counter/timers can be started and stopped by the processor, and their count values can be set to specific values or read by the processor.

The value in a counter incremented at a precise rate is directly proportional to the time elapsed since it started counting (or we can calculate the difference from when it was previously read). This can be used make time or frequency

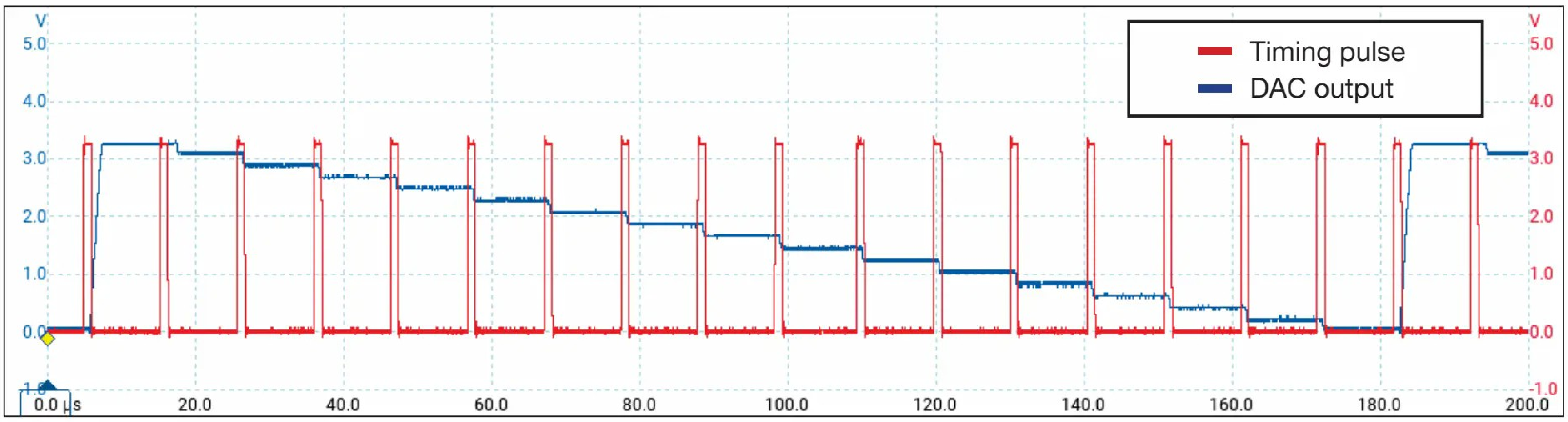


Fig.6: the DAC waveform with a sampling rate (frequency) of 96kHz.

```
Initialize DAC
Initialize Timer
Start DAC converting
Start Timer running
while(1)
{
  Reset Timer to zero
  Calculate new DAC value
  Update the DAC
  Wait until the Timer has reached
     the required count value
}
```

Listing 4: pseudocode for using a timer to control the DAC output loop rate.

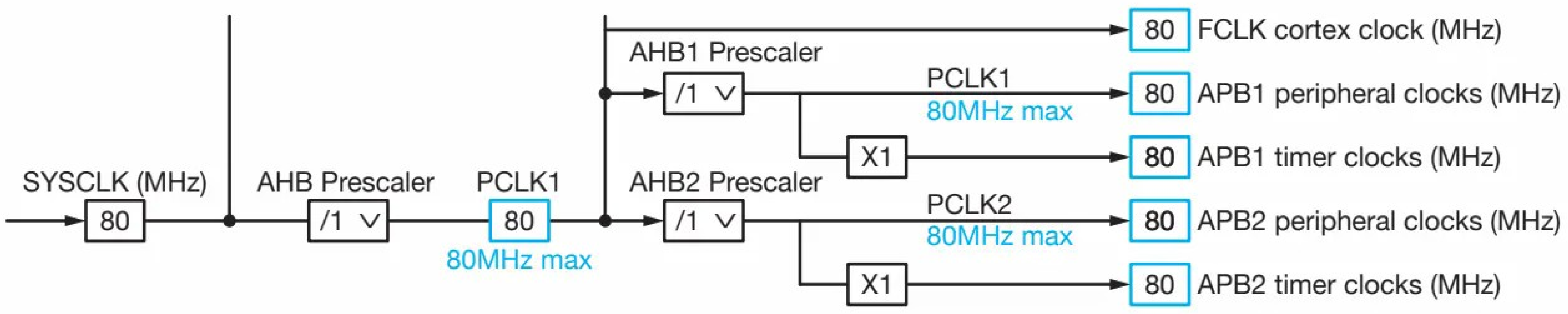


Fig.7: timer connections in the microcontroller's clock tree.

measurements of external events, or to control operations or events that must be accurately timed. Both one-off and repetitive timed operations can be controlled this way.

The previously mentioned function **`HAL_Delay()`** uses count difference values, but does not meet our requirements primary due to the fact that only millisecond timing is available.

To measure or control time with a timer, we need to set the frequency at which it is clocked. In STM32CubeIDE, this can be checked (and, if necessary, modified) in the Clock Configuration tab of the Device Configuration Tool. This provides a schematic of the complete clock system of the microcontroller, which is quite complex.

The microcontroller datasheet (**https://pemag.au/link/ac5i**) provides full details on the clock system.

The block schematic diagram in the Clock Configuration tab of the Configuration Tool shows that the timers are connected to the APB1/APB2 timer clocks that run at the 80MHz system clock (SYSCLK) frequency by default (APB is Advanced Peripheral Bus). Fig.7 shows the relevant part of the diagram.

The timer clock frequency can be reduced using the timer's prescaler (the APB1/APB2 prescalers in Fig.7). A prescaler is a circuit that reduces a high frequency to a lower one by an integer factor. The longer the period that needs timing, the lower the counter clock frequency needs to be for the counter to be able to count to a number within its range in the desired period.

The prescaler facilitates a very wide range of timing capability for the on-chip timers. For this project, dividing by 8 in the prescaler will make the counter count at 10MHz (one count every 0.1μs), which will facilitate easy calculation of delay values in the code.

There are several different timer types on STM32 microcontrollers (basic, general purpose, advanced etc). The microcontroller documentation provides details on this, allowing users to determine which best fits their requirements. We need a basic timer, and timer 6 (TIM6) on our microcontroller fulfils this role.

Before using the timer, we need to set it up using the Configuration Tool. To do this, go to the Pinout & Configuration tab of the Device Configuration Tool, click on Timers in the Categories list to expand it, then click on TIM6. Click on the Activated checkbox, which will cause the configuration settings to be displayed (Fig.8).

The frequency is divided by one plus the value entered into the prescaler configuration, so entering zero gives no frequency reduction (divide by 1). To set divide-by-eight, a value of seven is required in the "Prescaler (PSC – 16 bits value)" field of the configuration (Parameter Settings tab).

It is common practice for STM32CubeIDE users to write this as "8-1" (eight minus one) to make the required division value more obvious to read.

Using the timer

We will start with a simple use of the timer, similar to the previous example and outlined by the pseudocode in Listing 4. A timer is set to start counting at the beginning of the loop. As this is separate hardware, it is not affected by running other code, including the DAC update code. Once the DAC is updated, the code waits for the counter to reach a specific value before finishing the loop.

The time taken to execute each loop iteration, equal to the DAC sampling rate, is the time taken for the counter to reach the specified count value, so:

$$DAC\_sample\_period = counter\_period \times delay\_count\_value$$

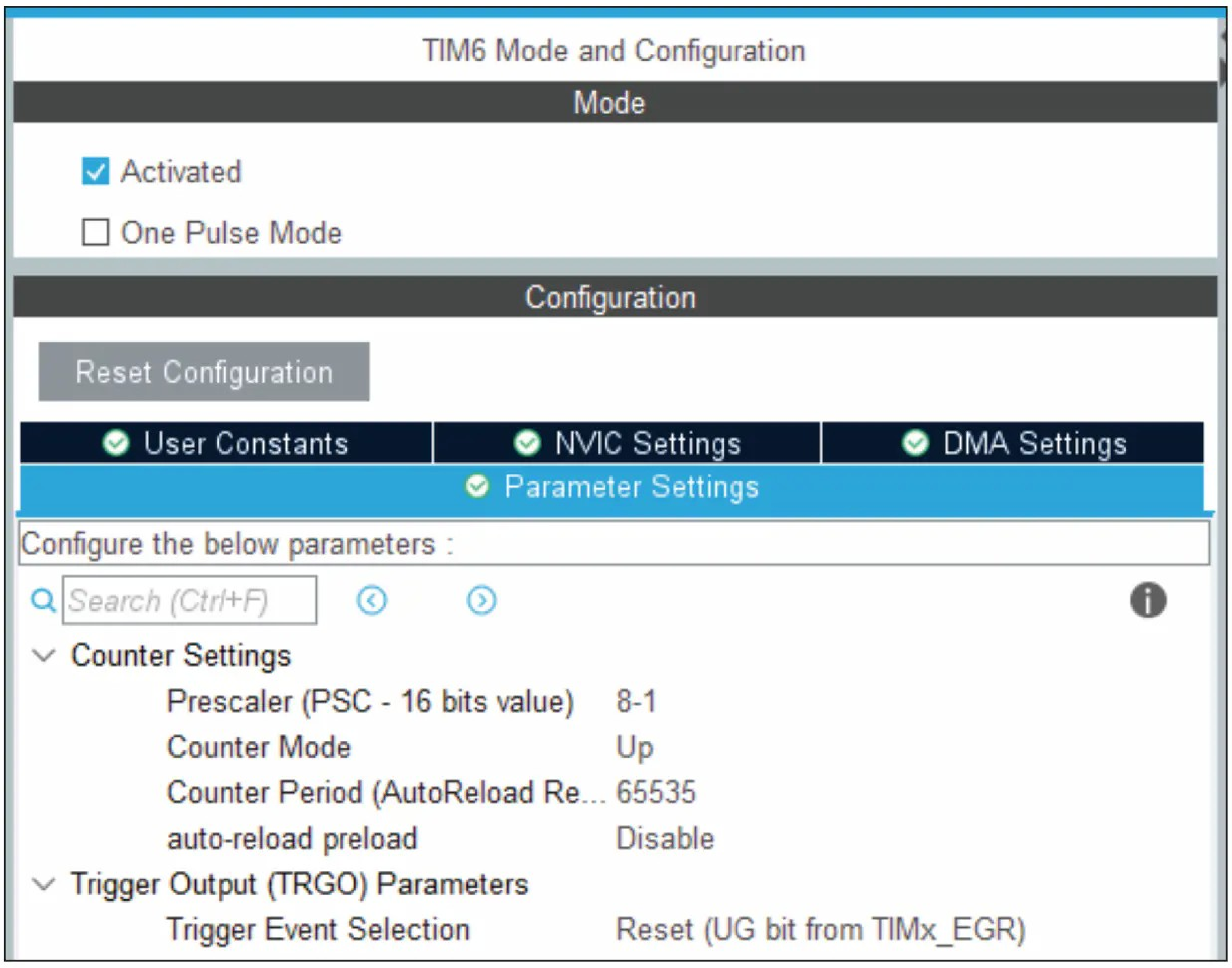
The sample period must be set to be longer than the time it takes to update the DAC and check the timer at least once; otherwise, the required delay time will already have passed before the waiting part is reached, and the timer will not be in control of how long the loop takes to iterate.

It is not a problem if the code has condition operations and may take different times on each iteration (unlike the for-loop delay) as long as the slowest case is fast enough. If follows that we do not need to know exactly how long the DAC update code takes, as the wait-for-timer step will control the loop iteration time.

However, the delay time must also be longer than the time the DAC output voltage takes to change, which is likely to be longer than the time taken to run the 'update DAC' code (as seen in the previous examples).

Timer-based DAC code

This example was created as a new



☑ Activated

☐ One Pulse Mode

Configure the below parameters :

Counter Settings

Prescaler (PSC - 16 bits value) 8-1

Counter Mode Up

Counter Period (AutoReload Re... 65535

auto-reload preload Disable

Trigger Output (TRGO) Parameters

Trigger Event Selection Reset (UG bit from TIMx_EGR)

Fig.8: the microcontroller's timer configuration settings in the IDE.

STM32CubeIDE project, with the pins and DAC configured as in the previous example and Timer 6 configured as just discussed. The DAC waveform was calculated in the same way as before, so the same variables are declared for this, along with `Time_Delay` for setting the required counter value in the waiting step (Listing 5, lines 61–64).

The timer is started in the User Code 2 section, along with the DAC. Timer 6 is referenced using the `htim6` handle in the same way as `hdac1` is used for the DAC (Listing 5, lines 126 & 127).

The main loop uses the same code as before to calculate the DAC values and set its output and timing pulse (Listing 5, lines 140–148). As indicated in the pseudocode, we need to start the counter from zero in each loop iteration. This can be done by writing zero directly to the count register (Listing 5, line 137).

To check if the counter has reached the required value for the delay, its current value can be read using a HAL function `__HAL_TIM_GET_COUNTER(&htim6)` (Listing 5, line 151).

For a 96kHz sampling rate, we need a sample period of 1 ÷ 96kHz = 10.417µs. Given that the counter increments at 0.1µs intervals, we should use a value of 104 for `Time_Delay`. However, experiments indicated that 101 gives the closest value at 96.7kHz. The DAC waveform is essentially the same as shown in Fig.6.

The loop delay is about 0.3µs longer than the timer value used. This is due to the time between the counter reaching its ultimate value and it being reset. During this time, the processor is executing the code that controls the loop iterations for the waiting and main loops.

Using a prescaler value of 8 makes the time increment nice round 0.1µs steps, but this limits the choice of frequencies, particularly for higher values.

Interrupts

This example makes use of a hardware timer, but not in the most efficient way. Software-based delays such as using for-loops or the `HAL_Delay()` function are referred to as blocking code – they tie up the processor when it could be doing more useful operations.

Although in the second example, the DAC is updated while the timer is independently running, we still have blocking code constantly checking the timer.

A better approach it to use an interrupt to automatically trigger some specific code to run (in this case, the DAC update) when the required time has elapsed (the DAC sampling time in this case).

In general, an interrupt is an input to a processor that causes it to stop what it is doing and respond to the interrupt event by running some specific code. After completing the interrupt code, called an Interrupt Service Routine (ISR), the processor returns to the previous activity exactly where it left off.

```
60   /* USER CODE BEGIN PV */
61   uint32_t DAC_Start = 4095; // Value at peak of sawtooth wave
62   uint32_t DAC_Value = 4095; // Current DAC output value
63   uint32_t DAC_Step = 255;   // Change in DAC value at each step (sample)
64   uint32_t Time_Delay = 101; // TIM6 timer delay count
65   /* USER CODE END PV */
…
125    /* USER CODE BEGIN 2 */
126    HAL_DAC_Start(&hdac1, DAC_CHANNEL_2);
127    HAL_TIM_Base_Start(&htim6);
128    /* USER CODE END 2 */
…
132    while (1)
133    {
134      /* USER CODE END WHILE */
135
136      /* USER CODE BEGIN 3 */
137      htim6.Instance->CNT = 0; // Reset counter
138
139      // Calculate next sawtooth waveform value for DAC
140      if (DAC_Value < DAC_Step)
141        DAC_Value = DAC_Start;
142      else
143        DAC_Value = DAC_Value - DAC_Step;
144
145      // Update the DAC and pulse the D8 pin to help measure timing
146      HAL_GPIO_WritePin(GPIOB, ARD_D8_Pin, GPIO_PIN_SET);
147      HAL_DAC_SetValue (&hdac1, DAC_CHANNEL_2, DAC_ALIGN_12B_R, DAC_Value);
148      HAL_GPIO_WritePin(GPIOB, ARD_D8_Pin, GPIO_PIN_RESET);
149
150      // Wait until the TIM6 counter reaches the time delay before iterating
151      while (__HAL_TIM_GET_COUNTER(&htim6) < Time_Delay);
152
153      }
154      /* USER CODE END 3 */
```

Listing 5: using a timer to control the DAC's sample update rate timing.

Interrupts can be triggered by any event that can be created or detected by hardware, either internal (on-chip) or external to the processor (such as a digital input pin changing state).

For timers, an interrupt can be generated after a specific elapsed time, causing the ISR code to run at that time. To produce a signal from a DAC, a hardware timer can be used to generate an interrupt at regular intervals (at the DAC sampling rate) with the ISR configured to set the DAC value.

Like other hardware setups, the Device Configuration Tool is used to control what conditions can cause interrupts in a design – for example, a timer can be set to cause an interrupt when it reaches a particular count value.

In a later article, we will see how data transfer from a memory buffer to a DAC can be achieved automatically without having to explicitly write to the DAC in our code. This is called direct memory access (DMA). We just have to periodically update the buffer with the DAC waveform; interrupts can be arranged to occur when this needs doing.

Callbacks

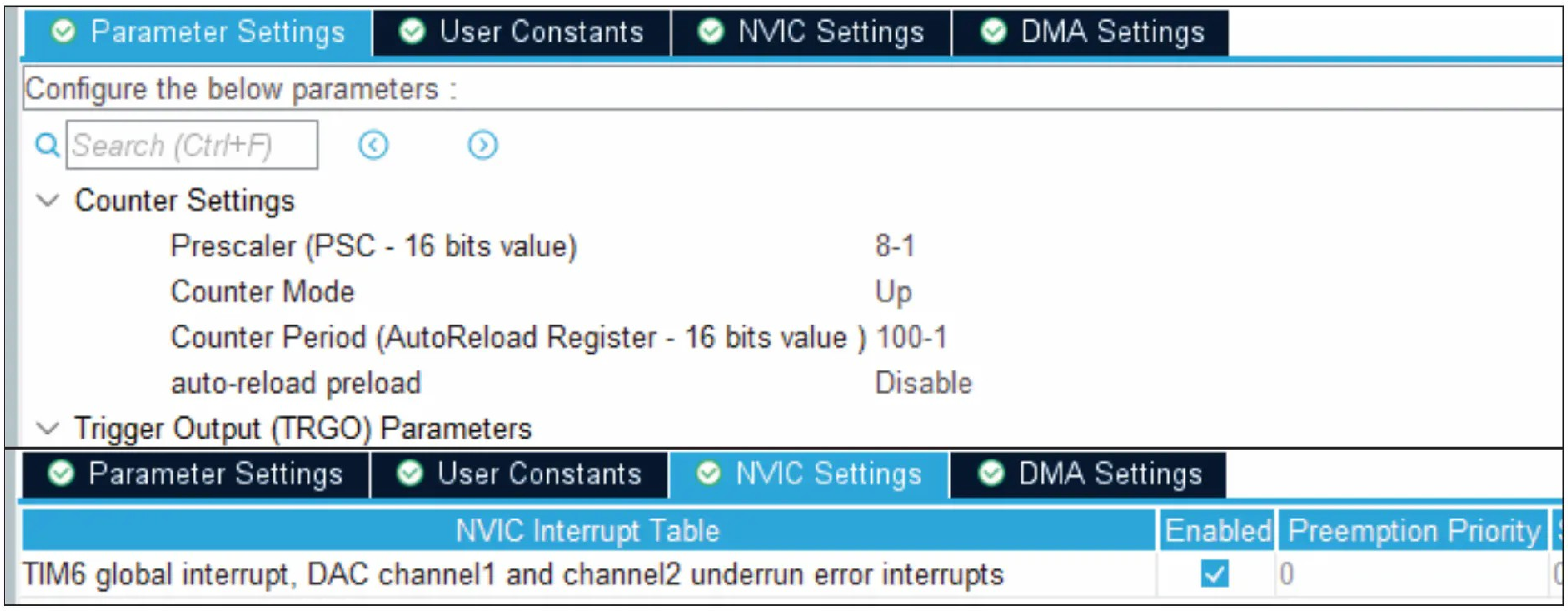
The STM32 code libraries include all the general code required to make ISRs work correctly (eg, to enable the processor to switch safely from what it was doing to the ISR and back). However, the libraries cannot contain the code the developer wants to run when a specific interrupt occurs.

To achieve this, the library defines special 'callback' functions that the user can define to handle their specific part of the ISR.

Callback functions are widely used in software in a situation where there is code that needs to do 'something' when a particular condition occurs, but at the time of writing the code managing the condition detection (eg, by the microcontroller library developer), it is not known what that 'something' is.

A mechanism is required that can later define what the 'something' is to run without changing the general condition/interrupt handling code. There are a few ways to do this; one is to define the name of a function that will get called when the 'something' needs to be done.

Effectively, the function already exists but does nothing; however, if another function with the same name is written (by the user) then it will be run instead. The user simply writes the required code in a function with the predefined name, and it automatically



Configure the below parameters :

Counter Settings

Prescaler (PSC - 16 bits value)	8-1
Counter Mode	Up
Counter Period (AutoReload Register - 16 bits value)	100-1
auto-reload preload	Disable

Trigger Output (TRGO) Parameters

NVIC Interrupt Table	Enabled	Preemption Priority
TIM6 global interrupt, DAC channel1 and channel2 underrun error interrupts	☑	0

Fig.9: the timer settings for interrupt use (a portion of the Parameter tab is at the top, while the NVIC tab is below it).

runs when the related condition (eg, an interrupt) occurs.

The STM32 HAL library provides many predefined callback functions. In this case, we want an interrupt after a time period has elapsed, as measured by a timer, so we use the function named **`HAL_TIM_PeriodElapsedCallback()`**. This function is called when various timers trigger an interrupt, so the code has to check which timer caused it to be called.

A timer handle is passed to the function as a parameter named **`htim`**. This can be used to find out which timer caused the callback to run. For example, the code **`if (htim == &htim6)`** will only run what follows that statement if it was Timer 6. Full details are shown in the example code later.

Configuring a timer for interrupts

The period of a timer can be set in the Device Configuration Tool and is referred to as the Count Period or the AutoReload Register (*ARR*) value. The counter elapsed interrupt occurs once every *ARR* counts of the counter. As discussed above, the clock period of the counter is given by the APBx timer clock (f_{APB}) divided by the prescaler (*PSC*) value.

Thus, the period for one count increment is $PSC \div f_{APB}$. The interrupt occurs every *ARR* counts, so the period between interrupt calls is $(ARR \times PSC) \div f_{APB}$. For example, knowing f_{APB} = 80MHz, if we set *PSC* = 8 (as previously) and *ARR* = 100 (entered as 100-1 in the configuration), the period between interrupt calls is (100 × 8) ÷ 80MHz = 10µs.

This will be the sampling period if the timer period elapsed ISR callback is used to update the DAC (a sampling frequency of 100kHz).

To set this up in the Device Configuration Tool, select TIM6, check that it is activated and has the prescaler value set to 8-1 as previously. In the Parameter Settings tab, set the Counter Period to 100-1, as shown in Fig.9. Leave other values as their defaults. In the NVIC (Nested Vectored Interrupt Controller) Settings Tab, enable the global interrupt for TIM6 by checking the Enable box (see Fig.9).

Configuring the timer with code

In the examples discussed so far, the timer is set up via the Device Configuration Tool. This is fine in some situations, but it might be useful to be able set the sampling frequency in user code (eg, if it needs to vary). We can look at the auto-generated code to find out how to configure the timer (or check the datasheet).

The function **`MX_TIM6_Init()`** is called before the main loop, along with various other hardware initialisation functions. The code of the function itself can be found about halfway through the auto-generated **`main.c`** file.

In the **`MX_TIM6_Init()`** function, various values are set via the **`htim6`** handle, based on the values entered via the Device Configuration Tool. For example, the period value of 100 discussed above is set using the code **`htim6.Init.Period = 100-1;`**

After this, **`HAL_TIM_Base_Init()`** is called to initialise the timer with the appropriate values.

The function returns a status value, which should equal **`HAL_OK`**. An error handling function, **`Error_Handler()`**, is called if there is a problem.

Similar code can be used again later (in user code) to change the timer settings. Code can also be written to calculate the counter parameter values from other data; for example, the period value can be calculated, based on having the required frequency (in kHz) and prescaler values, using **`Period = (80000.0/Prescaler)/Frequency;`** where 80000 is the 80MHz APBx timer clock frequency in kHz.

Timer-Interrupt DAC project code

This example was again created as a new STM32CubeIDE project. The configuration of the pins and DAC was as in the previous example, and Timer 6 was set up in the Configuration Tool as just described. The DAC waveform is calculated as in the previous examples, so uses the same variables (Listing 6, lines 61–63).

Other variables are declared for prescaler, period and frequency, to allow these to be manipulated in code as just discussed (Listing 6, lines 64–66). The period and frequency are floating-point values to ensure accurate calculation.

Before the main while() loop, in the User Code 2 section, the DAC is started as previously (Listing 6, line 128). Then the period configuration value for the timer is calculated as just described, and the prescaler and period values are set via the **`htim6`** handle, followed by reinitialisation of the timer.

```
/* USER CODE BEGIN WHILE */
while (1)
{
  /* USER CODE END WHILE */

  /* USER CODE BEGIN 3 */
  HAL_GPIO_TogglePin(GPIOB, LED2_Pin);
  HAL_Delay(500);
}
/* USER CODE END 3 */
```

Listing 7: blinking the LED simultaneously with the DAC waveform generation.

The timer is then started (Listing 6, lines 130–140), at which point interrupts will be start being generated at timer period intervals. In this example, the prescaler value is fixed for simplicity, but code could be written to set the prescaler to the most suitable value for a given frequency.

The DAC waveform and timing pulse code are the same as in previous examples, but these are now placed in the **`HAL_TIM_PeriodElapsedCallback()`** ISR in the User Code 4 section, near the end of the auto-generated **`main.c`** file (Listing 6, lines 750–767). As discussed above, the ISR checks that Timer 6 caused the interrupt (Listing 6, line 753) before running the DAC code.

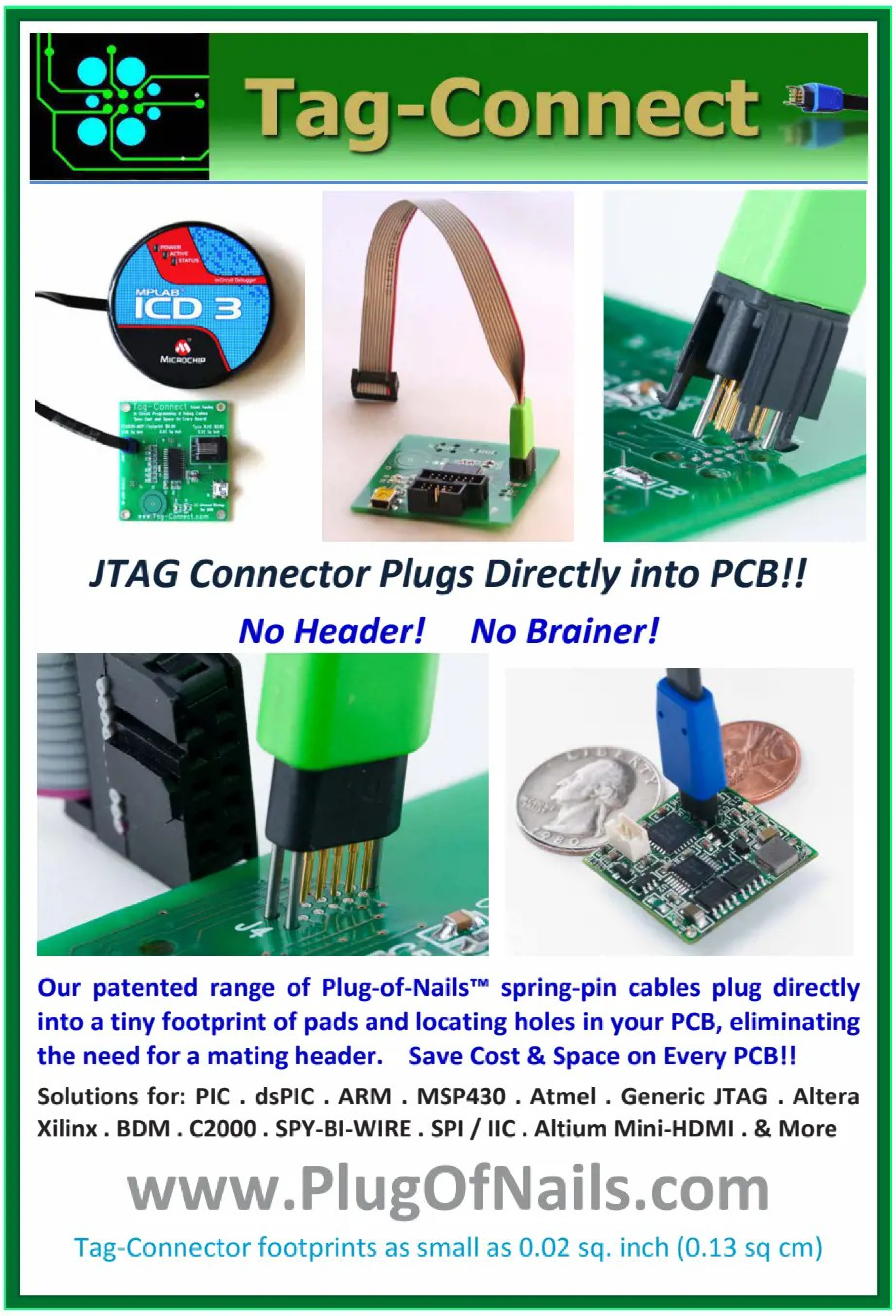
The DAC and timing pulse waveforms are again as in Fig.6, but now the measured frequency is very close to the required value (measured at 96.02kHz by the oscilloscope) due to the lower prescaler value.

The main while() loop does not contain any code in this version, but to illustrate the processor performing other operations when it is not serving the timer interrupt we can add LED blinking code to the main loop, as shown in Listing 7.

This toggles a digital output pin connected to one LED on the development board every half second (500ms), using the **`HAL_Delay()`** function. The LED blinking does not have any effect on the DAC's sampling frequency.

```
60   /* USER CODE BEGIN PV */
61   uint32_t DAC_Start = 4095; // Value at peak of sawtooth wave
62   uint32_t DAC_Value = 4095; // Current DAC output value
63   uint32_t DAC_Step = 255;   // Change in DAC value at each step (sample)
64   uint32_t Prescaler = 1;    // Timer prescaler setting
65   float Period;              // Timer period in us
66   float Frequency = 96.0;    // DAC sample frequency in kHz
67   /* USER CODE END PV */
…
127   /* USER CODE BEGIN 2 */
128   HAL_DAC_Start(&hdac1, DAC_CHANNEL_2);
129
130   // Calculate TIM6 counter period (ARR) from required DAC sample frequency
131   // Clock frequency is 80 MHz
132   Period = (80000.0/Prescaler)/Frequency; // F in KHz
133   htim6.Init.Prescaler = Prescaler-1;
134   htim6.Init.Period = Period-1;
135   // Reinitialise TIM6 with new values and start it
136   if (HAL_TIM_Base_Init(&htim6) != HAL_OK)
137   {
138     Error_Handler();
139   }
140   HAL_TIM_Base_Start_IT(&htim6);
141   /* USER CODE END 2 */
…
750  /* USER CODE BEGIN 4 */
751  void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
752  {
753    if (htim == &htim6)
754    {
755      // Calculate next sawtooth waveform value for DAC
756      if (DAC_Value <= DAC_Step)
757        DAC_Value = DAC_Start;
758      else
759        DAC_Value = DAC_Value - DAC_Step;
760
761      // Update the DAC and pulse the D8 pin to help measure timing
762      HAL_GPIO_WritePin(GPIOB, ARD_D8_Pin, GPIO_PIN_SET);
763      HAL_DAC_SetValue (&hdac1, DAC_CHANNEL_2, DAC_ALIGN_12B_R, DAC_Value);
764      HAL_GPIO_WritePin(GPIOB, ARD_D8_Pin, GPIO_PIN_RESET);
765    }
766  }
767  /* USER CODE END 4 */
```

Listing 6: using timer interrupts to control the DAC's sample update rate timing.



Tag-Connect

JTAG Connector Plugs Directly into PCB!!

No Header!   No Brainer!

Our patented range of Plug-of-Nails™ spring-pin cables plug directly into a tiny footprint of pads and locating holes in your PCB, eliminating the need for a mating header.   Save Cost & Space on Every PCB!!

Solutions for: PIC . dsPIC . ARM . MSP430 . Atmel . Generic JTAG . Altera Xilinx . BDM . C2000 . SPY-BI-WIRE . SPI / IIC . Altium Mini-HDMI . & More

www.PlugOfNails.com

Tag-Connector footprints as small as 0.02 sq. inch (0.13 sq cm)

ENIGMA
COMPONENT SHOP

www.enigma-shop.com

Electronic Components for the Hobbyist!

Cable & Wire • Capacitors • Connectors • Filters
Kits & Modules • Hardware & Cooling
and much more...

READY TO BE REWARDED?

Join our loyalty scheme and **earn points** on every purchase you make which can be **used for discounts** in the future.

Read our reviews on Trustpilot

PROJECT BY TIM BLYTHMAN

This handy, portable, rechargeable device combines a clock, timer and stopwatch and can display different time zones. It has an internal crystal and incorporates a WiFi time source, so it is always accurate, even if a leap second occurs.

DOWN
UP
OK
MODE
SYDNEY
B=3.9V
2:00:52
P
TUE 21 MAY 2024
SILICON CHIP
14A_Y71_231117
COMPACT
CLOCK & TIMER

COMPACT OLED CLOCK/TIMER

YOU MIGHT THINK THAT WHAT THIS CLOCK/TIMER does could easily be done by an app on a smartphone, and you are probably right. In fact, many of the ideas we have for electronic projects are dismissed because they could be 'just an app'.

However, there is a good reason to make the Clock/Timer a separate device. My wife runs a business where she needs to keep track of time spent with clients.

Using a phone app to do that tends to drain the phone's battery and makes it difficult to use the phone for other purposes.

So, a separate device that can keep track of time has its place. The Compact OLED Clock and Timer also makes it easier to keep track of time in different time zones. This is another handy feature if you arrange appointment times with people in different locations.

It has an alarm feature that is tied to a 'home' timezone. This means that if you are travelling, you can be alerted each day at the same time in that zone, even if you are using the Clock to see the local time in a different time zone. This feature is notably absent from most clock apps.

There are countdown timer and stopwatch functions that can work in the background. For example, you can set the countdown timer running and then switch to the clock or stopwatch. The timer will still alert you when it is finished.

Internally, time is kept by a watch crystal. An integrated WiFi time source is also used to keep the time updated and trim out any crystal errors. Time is kept to the nearest second, so you should never be more than a few seconds out.

Compact case

The clock fits in a UB5 Jiffy box and its blue front panel is the back of the PCB that hosts most of the components. Its user interface comprises a small OLED screen that shows through a hole in the PCB/panel, and four pushbuttons to control its functions. As most of the parts are surface-mounted, there are no solder joints on the front.

We think this size and shape work well for a clock. The box can sit on its edge with the display clearly visible, but it is also unobtrusive.

While the form factor is similar, this design uses a different processor from the Pico Audio Analyser, and the circuitry is quite different.

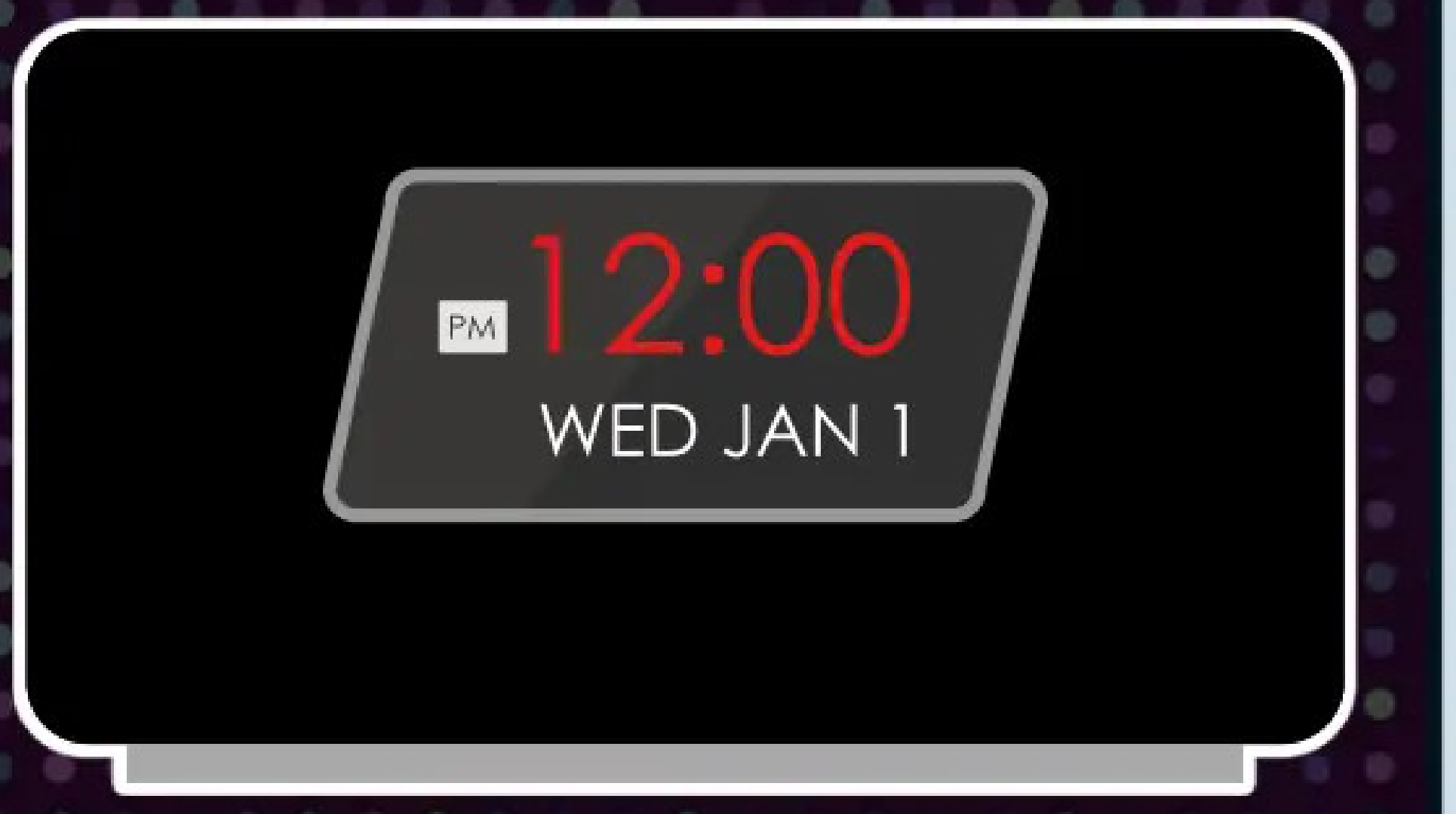
Circuit details

The complete circuit diagram of the Clock/Timer is shown in Fig.1. Instead of a Pico microcontroller board, it is controlled by a PIC16F18146 microcontroller (IC1). Since this IC is capable of low-current operation, we can dispense with the complexity of providing an on/off switch.

Microcontroller IC1, in combination with 32768Hz crystal and its two 4.7pF load capacitors, is responsible for timekeeping. IC1's oscillator can remain operating even when it is in deep sleep

OLED Clock & Timer Features & Specifications

- Clock with multiple time zones
- Automatic daylight saving adjustments
- Alarm with day and repeat options
- Countdown timer up to 99 hours
- Stopwatch up to 99 hours
- Rechargeable 600mAh battery
- Battery charging and status display
- OLED screen with adaptive brightness
- Resolution: one second
- Crystal timekeeping backed by integrated WiFi time source
- Current draw: 15-20mA during operation, 5mA with screen off
- Current draw during WiFi operation: up to 80mA (typically for 30s per day)



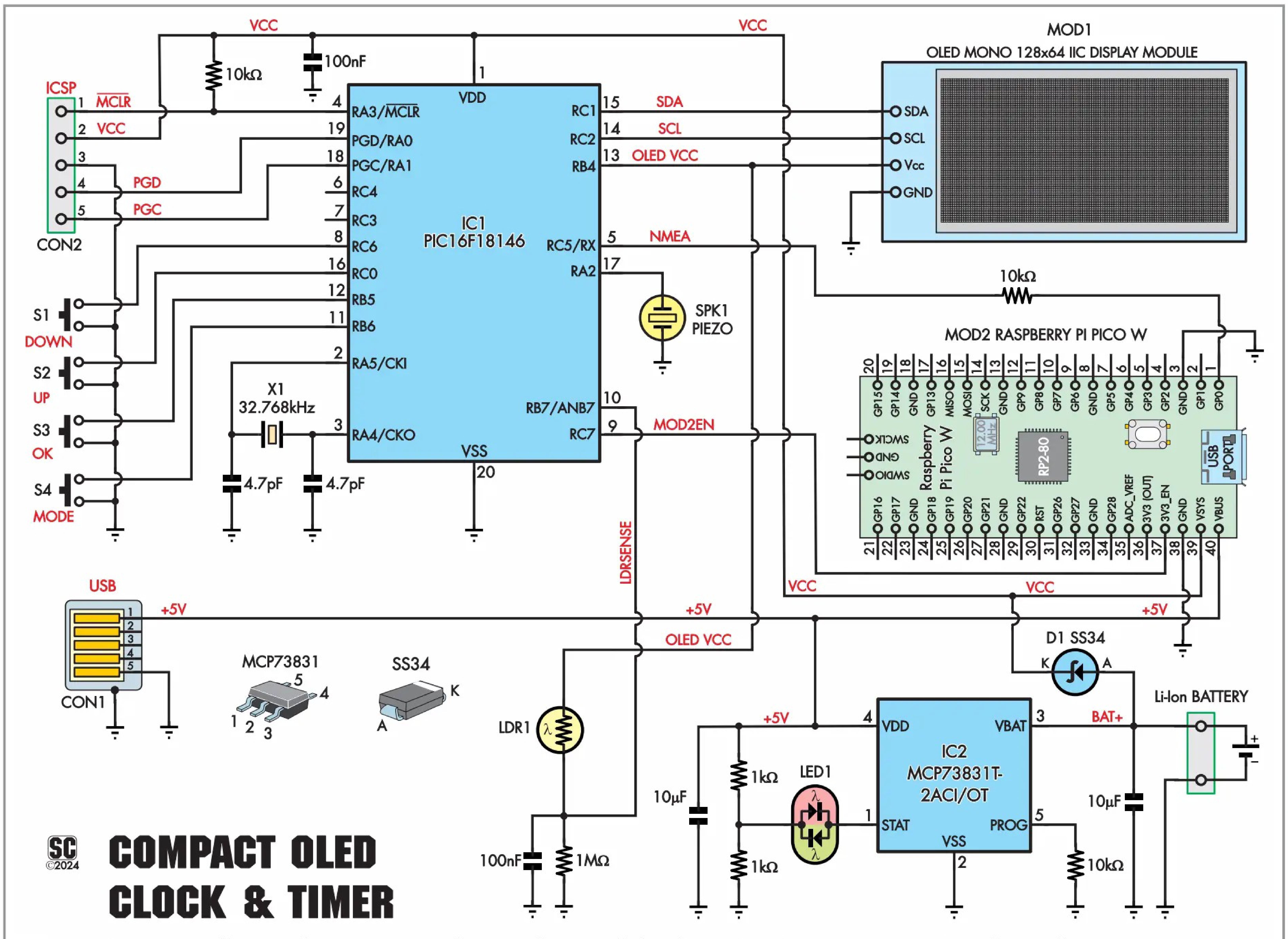

VCC
10kΩ
100nF
ICSP
1 MCLR
2 VCC
3
4 PGD
5 PGC
CON2
S1 DOWN
S2 UP
S3 OK
S4 MODE
X1 32.768kHz
4.7pF
4.7pF
IC1 PIC16F18146
VDD
RA3/MCLR
PGD/RA0
PGC/RA1
RC4
RC3
RC6
RC0
RB5
RB6
RA5/CKI
RA4/CKO
VSS
RC1
RC2
RB4
RC5/RX
RA2
RB7/ANB7
RC7
SDA
SCL
OLED VCC
NMEA
MOD2EN
SPK1 PIEZO
MOD1
OLED MONO 128x64 IIC DISPLAY MODULE
SDA
SCL
Vcc
GND
10kΩ
MOD2 RASPBERRY PI PICO W
Raspberry Pi Pico W
RP2-B0
USB PORT
SWCLK
GND
SWDIO
LDRSENSE
USB
+5V
CON1
MCP73831
SS34
LDR1
100nF
1MΩ
+5V
OLED VCC
VCC
D1 SS34
K
A
Li-Ion BATTERY
BAT+
10μF
1kΩ
1kΩ
LED1
IC2 MCP73831T-2ACI/OT
VDD
VBAT
STAT
PROG
VSS
10μF
10kΩ
SC ©2024
COMPACT OLED CLOCK & TIMER

Fig.1: microcontroller IC1 keeps time with crystal X1 and displays it on OLED screen MOD1 by updating it over an I²C serial bus. IC1 can also control the power supply to all other components, keeping the idle current low. The battery is kept charged by IC2, which also drives LED1 to display the charge status.

power-saving mode so that it continues to keep track of the passage of time.

IC1 also drives OLED screen MOD1 via an I²C serial bus to update the display. The display's power supply comes from digital output RB4 of IC1 (pin 13), so it is powered down when not in use by pulling that pin low or powered by bringing it high. The I²C bus pullup resistors are on MOD1; IC1 uses 'bit-banging' to drive SDA and SCL since high-speed data transmission is not required.

MOD2 is a Raspberry Pi Pico W board programmed as a WiFi time source with an NMEA output compatible with GPS modules. We previously described how that works in the June 2024 article on a Wi-Fi Time Source for GPS Clocks. Important to the operation of the circuit is that the Pico W has a schottky diode between its VBUS pin, pin 40 (anode) and its VSYS pin, pin 39 (cathode).

This diode feeds microcontroller IC1 when USB power is available since it will supply a higher voltage than the battery via schottky diode D1. This arrangement removes the load from the battery while it is charging, allowing it to charge fully.

The remainder of IC1's various I/O pins manage the clock functions and user interface. A 10kΩ resistor pulls up IC1's reset pin 4 to allow normal operation, except when a programmer is connected at optional ICSP (in-circuit serial programming) header CON2.

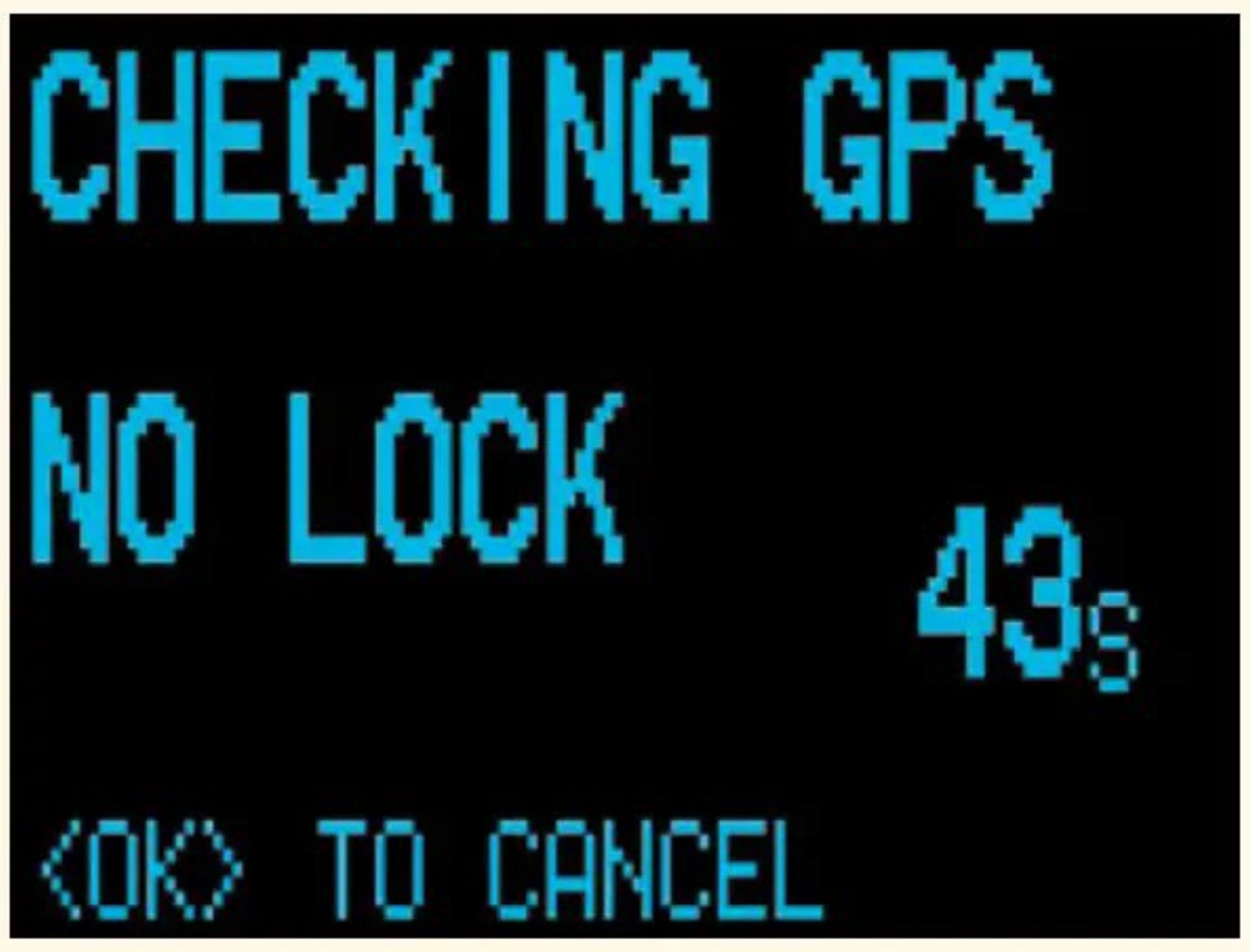
Four tactile pushbuttons, S1-S4, connect to pins 8, 16, 12 and 11 of IC1. These pins are set to have internal pullup currents, so the closure of the pushbuttons can be detected when the pin is pulled to ground. Pin 17 is connected to piezo transducer SPK1 to create alarm sounds.

Pin 13 of IC1 also powers a divider formed by the 1MΩ resistor and LDR1. The 100nF capacitor smooths the resultant voltage and provides a low-impedance input to IC1's ADC (analog-to-digital converter), sampled at pin 10 to measure the ambient light level.

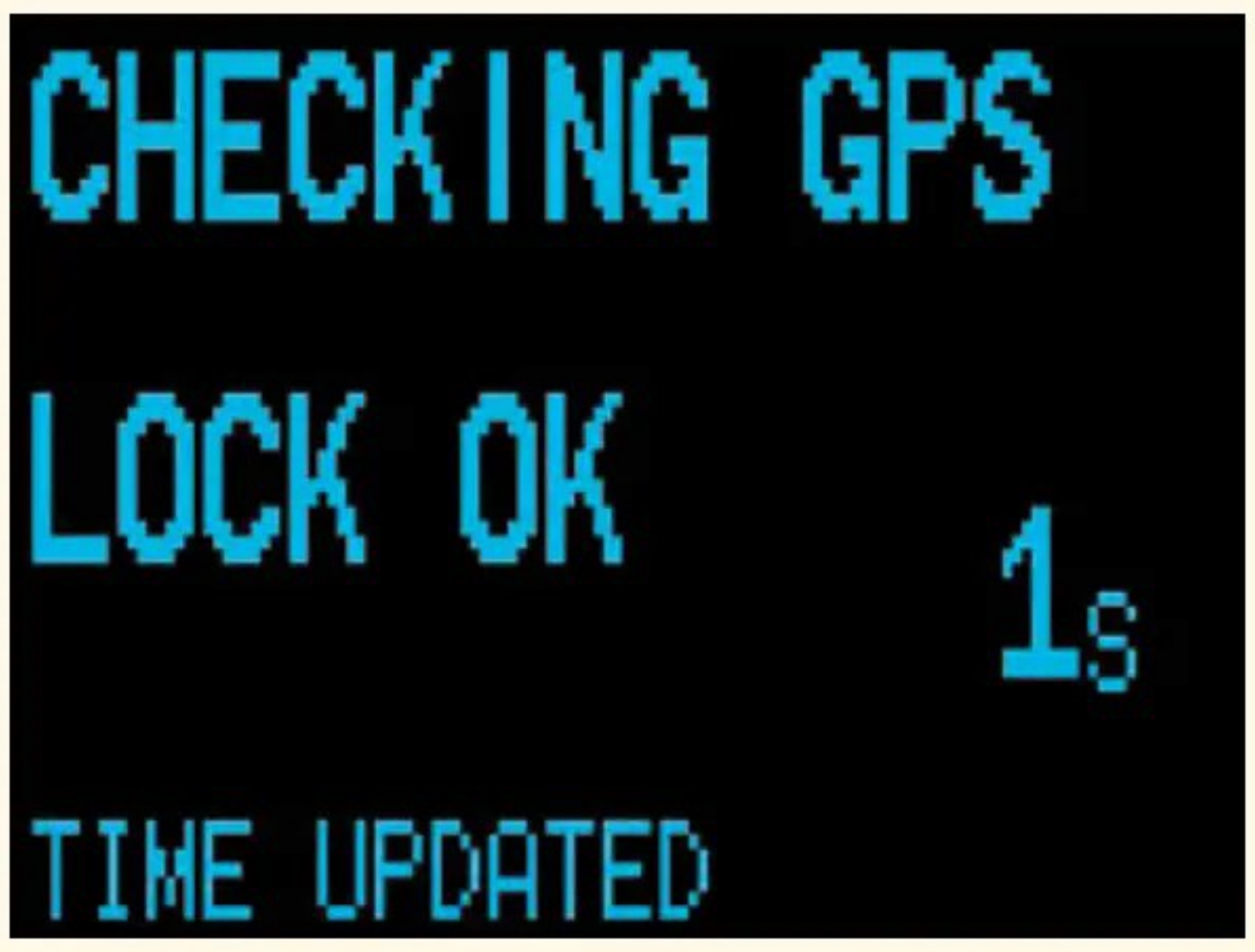
Pin 9 connects to the 3V_EN pin of MOD2. When this is pulled low by IC1, the 3.3V regulator on the Pico W is disabled and MOD2 is shut down. If it is allowed to float, it is weakly pulled up by the Pico W so it can operate. IC1 can thus choose to enable the time source only when needed.

The NMEA data stream from pin 1 on MOD2 is fed to pin 5 of IC1 via a 10kΩ resistor. Software running on IC1 decodes this data, including the time the time source has obtained via NTP. By comparing an internal 2.048V reference to its supply voltage, IC1 can also monitor the battery level or note that USB power is being supplied.

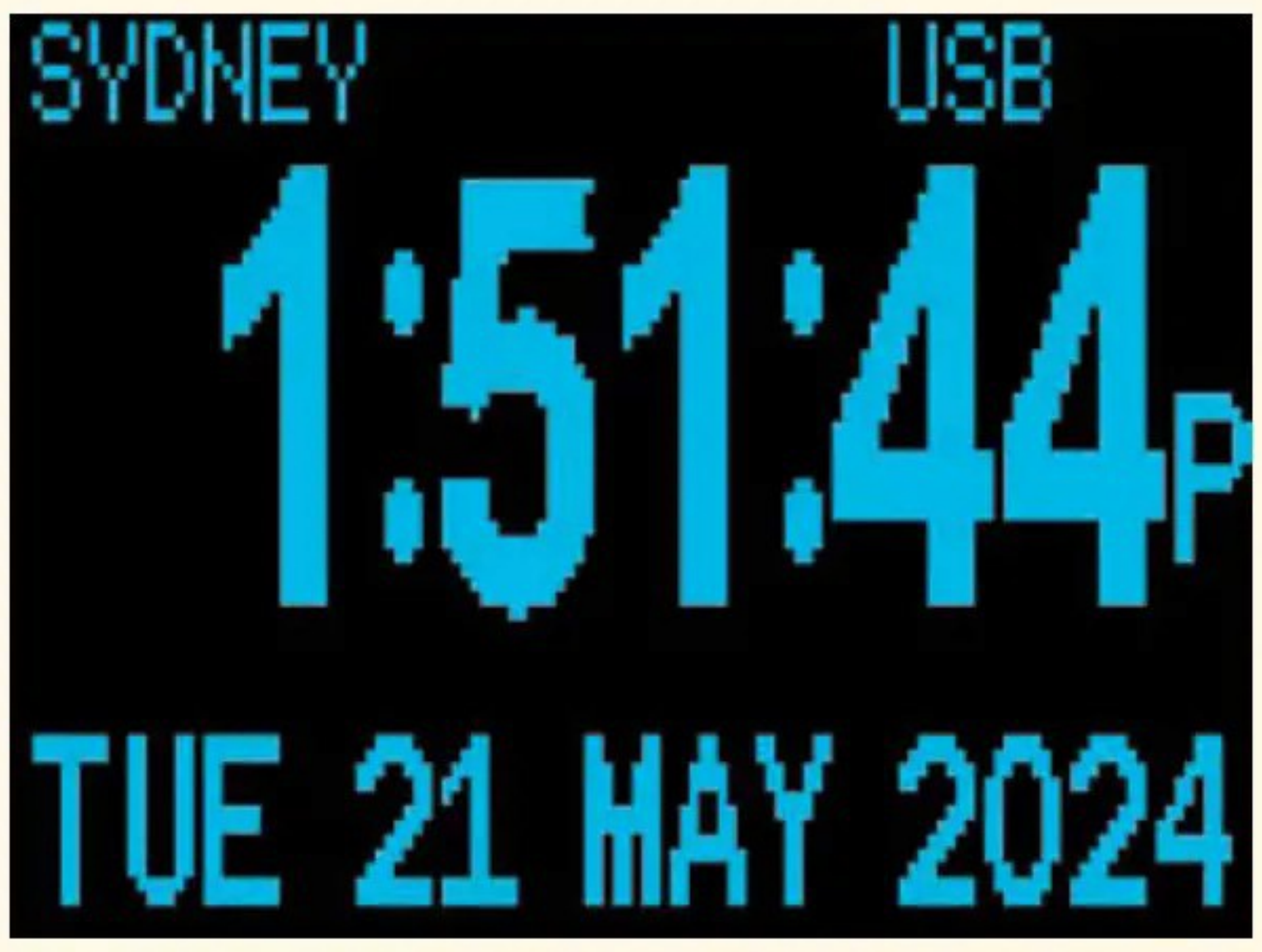
IC1 can shut down all of the surrounding circuitry by bringing its pins 9 and 13 low. The normal operating current draw is dominated by the OLED


CHECKING GPS
NO LOCK
43s
<OK> TO CANCEL

Screen 1: the initial screen; if you see the "NO DATA" message for more than a few seconds, check that the WiFi time source's LED is on or flashing. Once the time has been acquired, check that IC1 has shut it down. The battery life will be severely affected if the Pico W does not shut down.


CHECKING GPS
LOCK OK
1s
TIME UPDATED

Screen 2: this will briefly appear to show that the time has been updated. The Clock/Timer can be powered from the Pico W's USB socket, allowing you also to use the time source's USB interface for debugging. The 5V lines of the sockets are joined, so don't plug into both simultaneously.


SYDNEY USB
1:51:44P
TUE 21 MAY 2024

Screen 3: the Clock mode display. The default time zone is Sydney (the same as Melbourne, Canberra and Hobart). To access the settings, press and hold the MODE button until SETTINGS appears on the screen. All settings are kept in EEPROM and generally take effect immediately.

module, except for the brief periods when MOD2 is enabled.

Mini Type-B USB socket CON1 provides 5V power to the circuit. It goes directly to IC2, an MCP73831 Li-ion battery charging IC. 10µF bypass/filter capacitors are provided for its input and output, while the 10kΩ resistor on its PROG pin sets the battery charge current to 100mA. The battery is connected to the BAT+ and BAT– pads.

IC2 also provides a status indication at its STAT pin. Bicolour LED1 connects between the STAT pin and a pair of 1kΩ resistors between the 5V rail and ground. The STAT pin is low during charging and the red LED is driven. When charging is complete, the STAT pin goes high, allowing the green LED to light.

When 5V power is unavailable, the STAT pin is high-impedance and LED1 does not light. The power from the battery feeds IC1 via schottky diode D1. IC1 is powered at pins 1 and 20, with the standard 100nF bypass capacitor across them.

PCB arrangement

We've crammed an awful lot into a small enclosure, so we've opted for some creative assembly options. The pushbuttons and OLED display are reverse-mounted to protrude or show through the PCB that also forms the enclosure's front panel.

The LDR peeks through a hole in the front of the case too, while the LED shines through the PCB substrate from the back of the panel. We've used the copper layer and solder mask to create an outline resembling a battery icon.

We have designed the front panel PCB to sit over the edge of the enclosure rather than recess into it. This makes the Clock slightly deeper, giving more room for the battery and other components.

While we generally use USB-C sockets for power these days, we have stuck with a mini Type-B USB socket here to save a little more space; the USB-C sockets require two extra resistors to communicate the role of the device.

The various headers connect via surface-mounting pads, allowing wires to connect to devices in the space behind the PCB. The battery and speaker are both on flying leads to allow this.

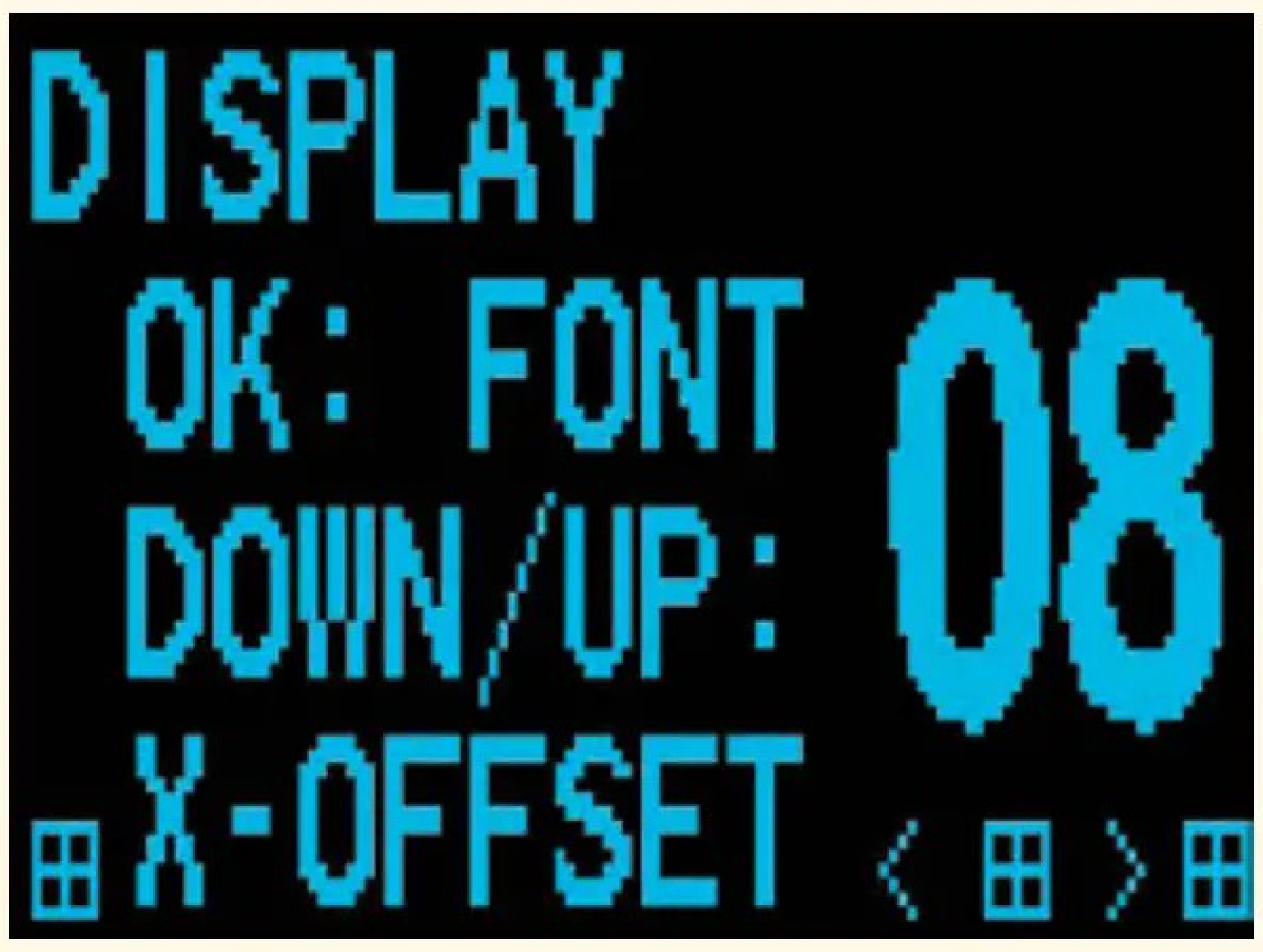
Parts List – OLED Clock and Timer

1 double-sided PCB coded 19101231, 83 × 53mm
1 UB5 Jiffy box (83 × 53 × 30mm) **[eg, Mouser 563-CU-1941]**
1 single AA cell holder with flying leads
1 14500 (AA-sized) Li-ion rechargeable cell with nipple (LiFePO$_4$ type recommended)
1 1.3-inch (33mm) OLED module (MOD1) **[Silicon Chip SC5026 or SC6511]**
1 Raspberry Pi Pico W programmed as WiFi Time Source for GPS Clocks (MOD2) **[Firmware: siliconchip.com.au/Shop/6/188]**
4 reverse-mount SMD tactile switches (S1-S4) **[Adafruit 5410]**
1 SMD mini-USB socket (CON1)
1 5-pin male header, 2.54mm pitch (CON2; **optional**, for ICSP)
2 4-pin male headers, 2.54mm pitch (for MOD2)
1 single-pin header (for MOD2)
1 100kΩ (light) to 10MΩ (dark) 5mm LDR (LDR1)
1 32768Hz watch crystal (X1)
1 passive piezo element (SPK1) **[eg, DigiKey 433-PT-1306T-ND]**
1 small tube of neutral-cure silicone sealant or similar
4 small self-adhesive rubber feet (**optional**)

Semiconductors
1 PIC16F18146-I/SO microcontroller programmed with 1910123A.HEX, SOIC-20 (IC1)
1 MCP73831-2ACI/OT Lithium battery charge regulator, SOT-23-5 (IC2)
1 SS34 40V 3A schottky diode, DO-214 (D1)
1 bi-colour red/green 3mm LED (LED1)

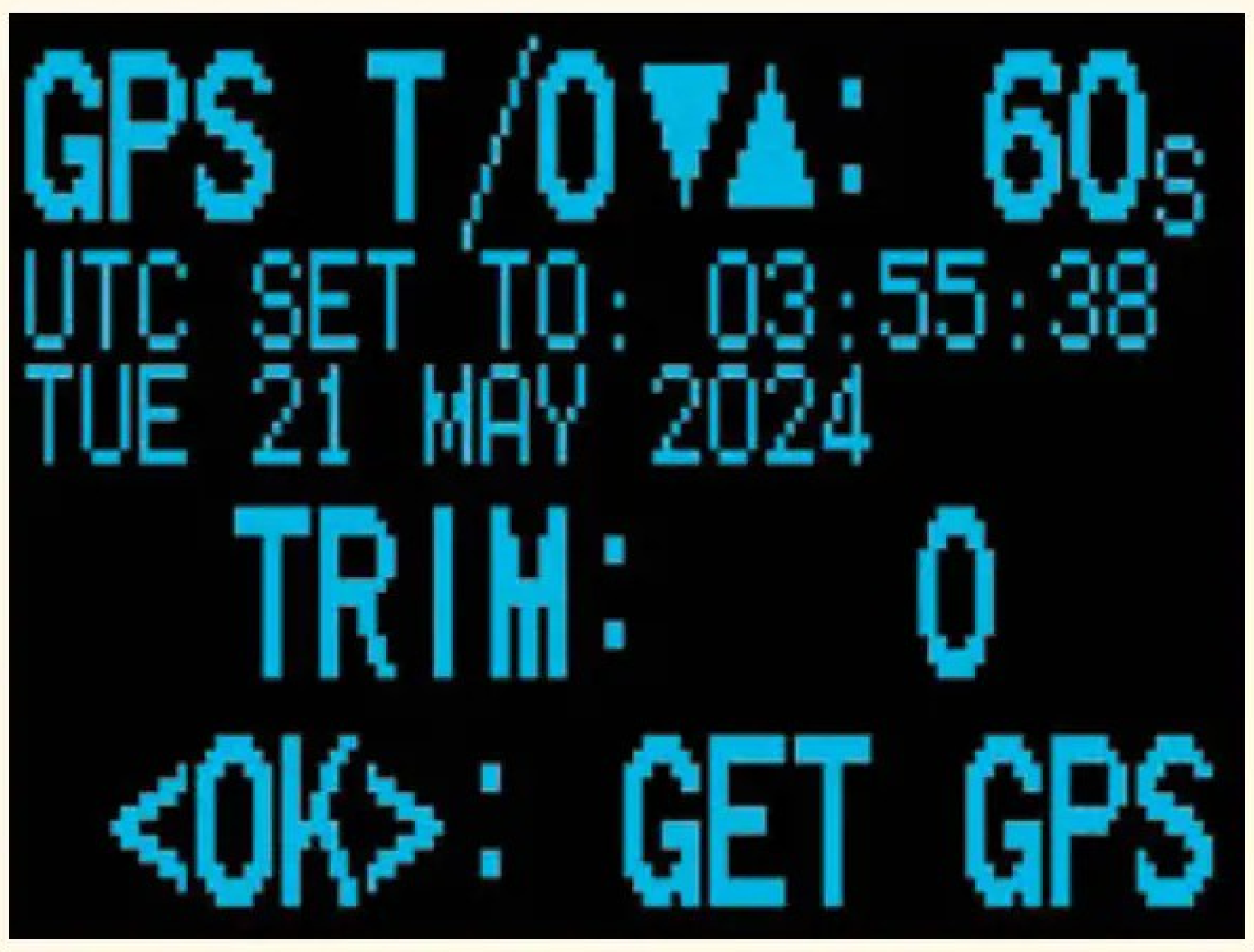
Capacitors *(all M3216/1206 size, X7R ceramic unless noted)*
2 10µF
2 100nF
2 4.7pF C0G (to suit crystal X1)

Resistors *(all M3216/1206 size, 1% ⅛W)*
1 1MΩ
3 10kΩ
2 1kΩ



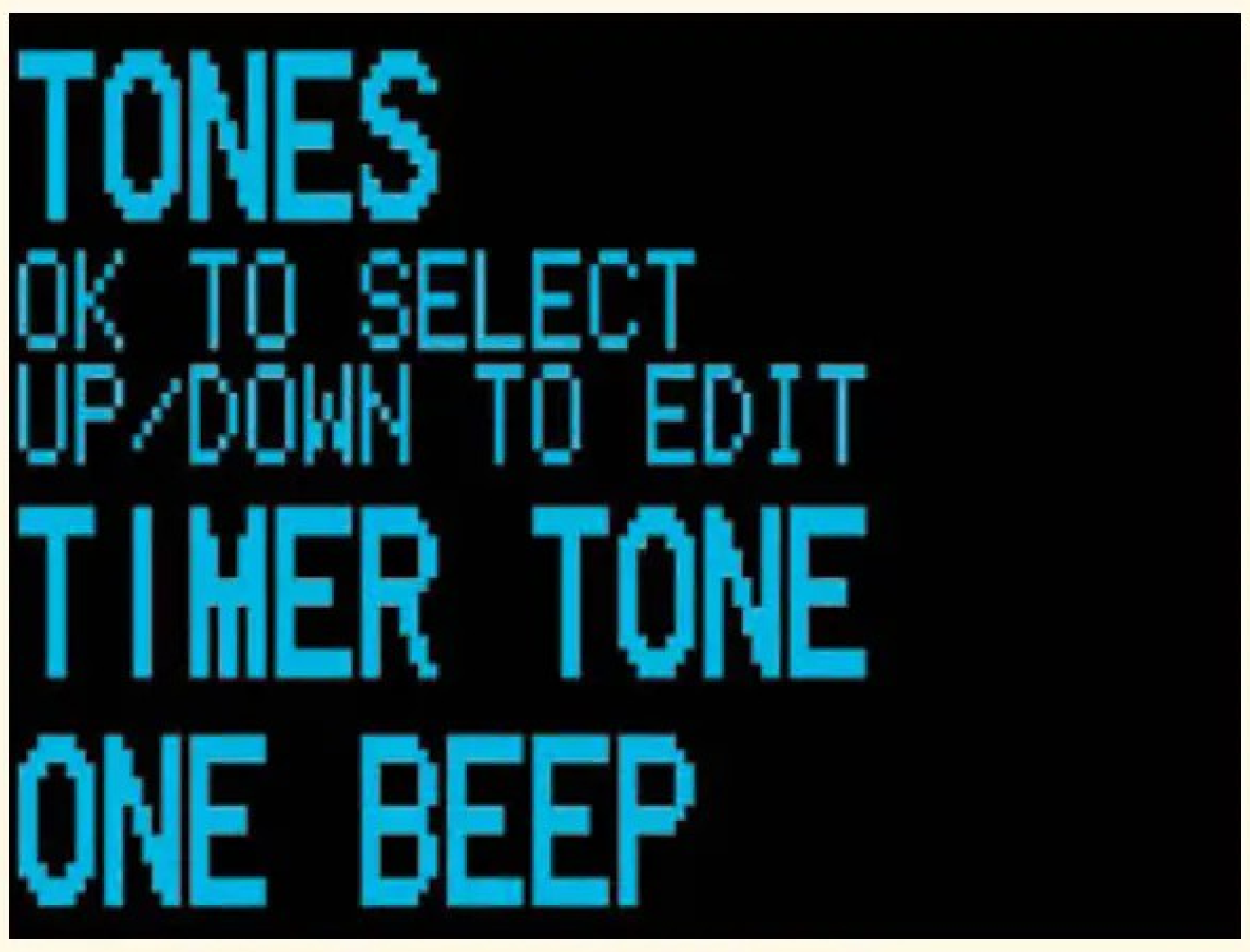
DISPLAY
OK: FONT
DOWN/UP: 08
⊞X-OFFSET ‹⊞›⊞

Screen 4: the OK button will cycle through the available fonts used for all large time displays. The UP and DOWN buttons trim the horizontal position of the display. Adjust the position until the box characters in both lower corners look the same as the one between the arrows.



GPS T/O▼▲: 60s
UTC SET TO: 03:55:38
TUE 21 MAY 2024
TRIM: 0
<OK>: GET GPS

Screen 5: MODE cycles between the SETTINGS pages. GPS refers to the time source; its maximum runtime can be set on this page. You can manually trigger a time update with the OK button. The TRIM value is zero initially but will update as the timekeeping is adjusted daily.



TONES
OK TO SELECT
UP/DOWN TO EDIT
TIMER TONE
ONE BEEP

Screen 6: test tones are played while this screen is showing. Press OK to toggle between the alarm clock tone and the countdown tone, then use the UP and DOWN buttons to choose which tone to use for each. If you don't hear a tone, there may be a problem with your piezo speaker.

The Pico W only needs connections on a handful of its pins; it is mounted behind the OLED module. The design of the time source puts all of its active pins at one end, which helps everything fit into the case.

Software

The watch crystal is used by a timer on IC1 to generate an interrupt once every second, making accurate timekeeping a priority. Every second, the clock is advanced; if the timer or stopwatch is active, they are also updated.

It keeps track of time internally as UTC (universal coordinated time) and calculates offsets based on the time zone and daylight saving status. While the inbuilt time zones are for Australia and New Zealand, since the designer lives in Australia, it also has a custom timezone that can be set to any time zone that is a multiple of 15 minutes from UTC/GMT (we aren't aware of any that are not).

The clock can display the current time in any of the time zones by selecting them. A 'home time zone' is selected, which is used to check the alarm. Every 24 hours, the WiFi time source is activated and the time is checked and updated (if necessary).

The Clock/Timer also checks how much drift has occurred and provides an internal correction for up to 24 seconds of drift per day. Watch crystals are typically well within that tolerance.

The WiFi time source can also be manually activated. A switch in the settings menu allows the Pico W that acts as the WiFi time source to be powered up. This can be handy for reprogramming the Pico W or changing the WiFi time source settings.

With a small amount of flash memory spare in the chip, we have added alternative fonts to provide some novelty to the main timekeeping display. There are also six different alarm tones, so you can choose your preferred alert sounds for the clock alarm and countdown timer.

These sounds are provided by combining a PWM signal with a UART (serial data) signal through the CLC peripheral. The rise and fall of the serial data modulates the signal, giving different tone patterns. Once the pattern is activated, it plays with no further processor input.

The details of the software operation and user interfaces will be discussed later.

Programming the chips

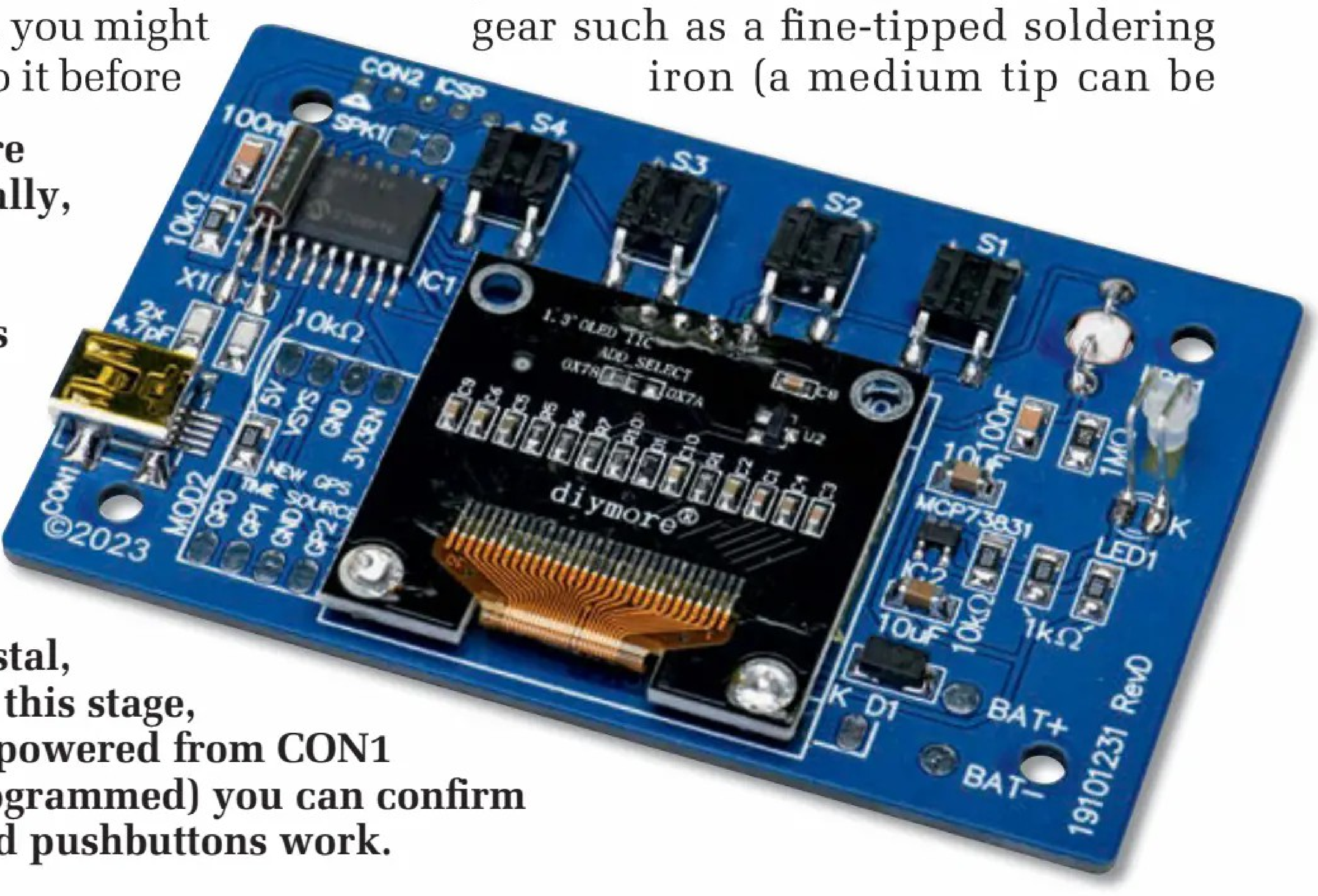
If your PIC16 microcontroller (IC1) is not programmed, you might find it easier to do it before soldering the chip to the board if you have the right gear. Our PIC Programming Adaptor article from September 2024 includes examples of SMD-to-DIP adaptors that can be used to do this.

Otherwise, you will have to make a temporary connection to the CON2 ICSP header after the chip is installed. You can see a header in some of our photos; this is what we fitted to CON2 to help with repeated programming during software development.

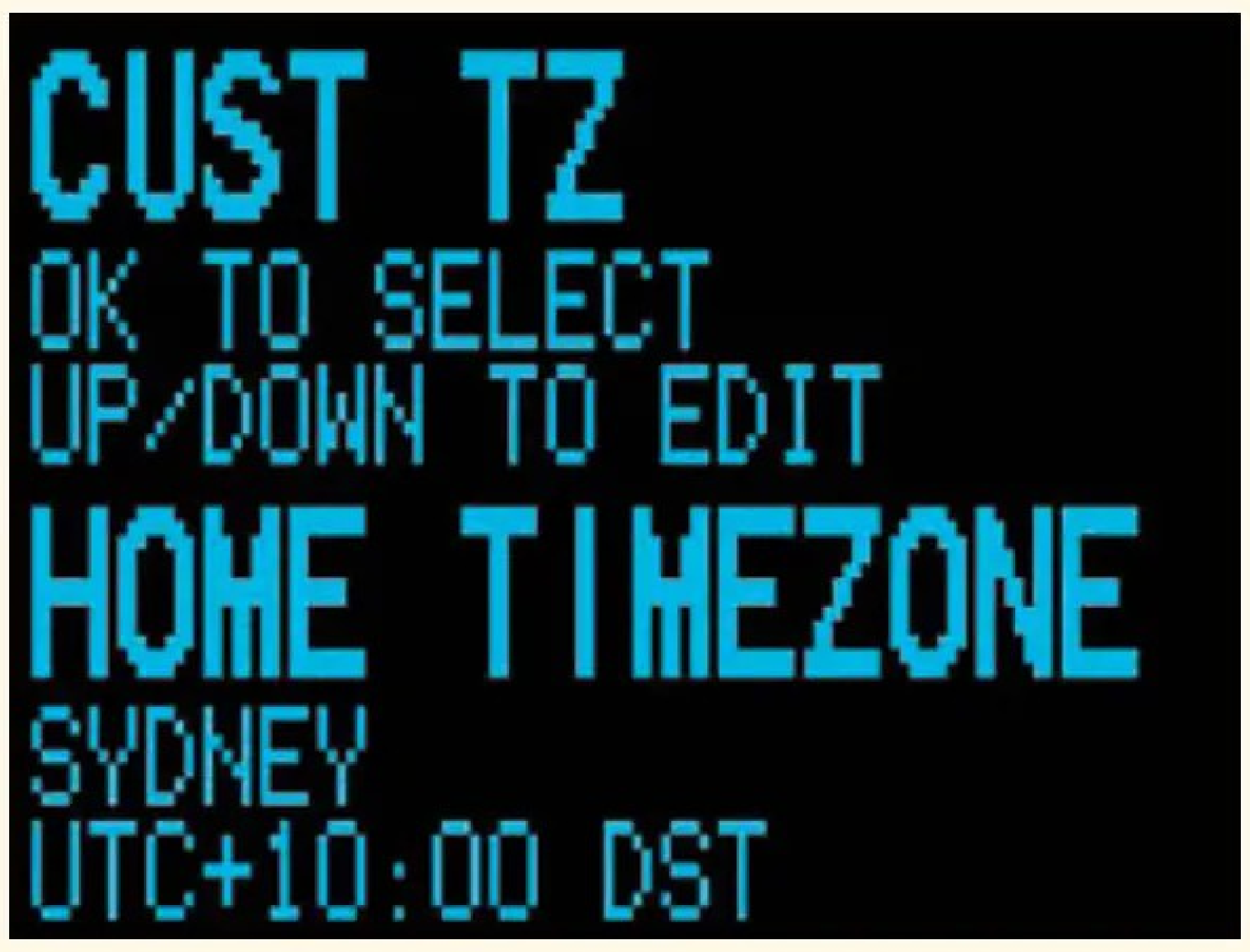
Programming the Pico W module can be easily done before or after soldering it. Simply connect it to a computer using a standard USB cable. See the panel on setting up the WiFi time source for more details.

Construction

The Clock/Timer is built on a double-sided PCB coded 19101231 that measures 83 × 53mm. The design necessitates surface-mount construction, so you will need the usual surface-mount gear such as a fine-tipped soldering iron (a medium tip can be

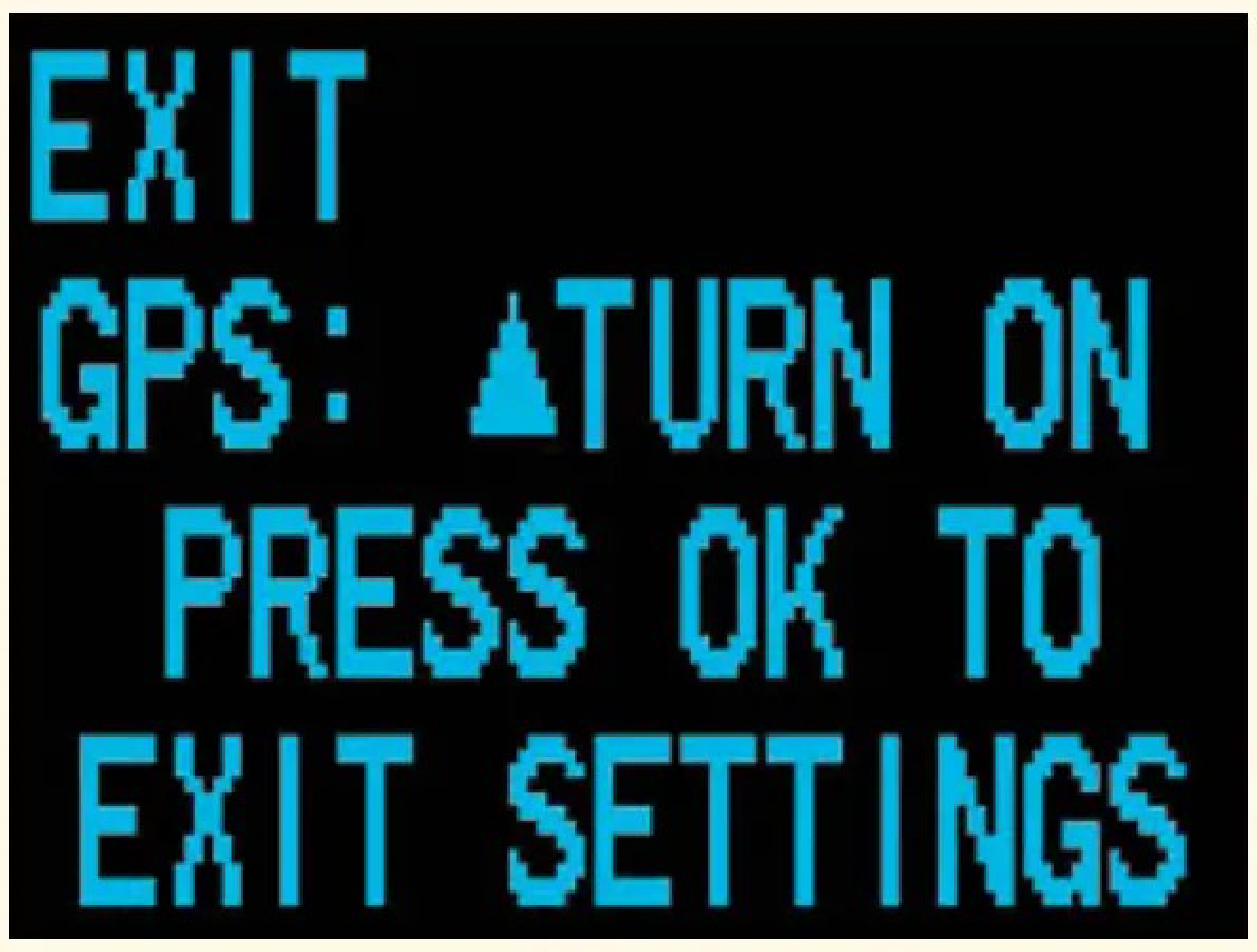


The SMD parts are fitted conventionally, although we recommend splaying the leads of S1-S4 so their stems project more through the panel. Note how we've fitted leaded parts like the crystal, LDR and LED. At this stage, the board can be powered from CON1 and (with IC1 programmed) you can confirm that the OLED and pushbuttons work.



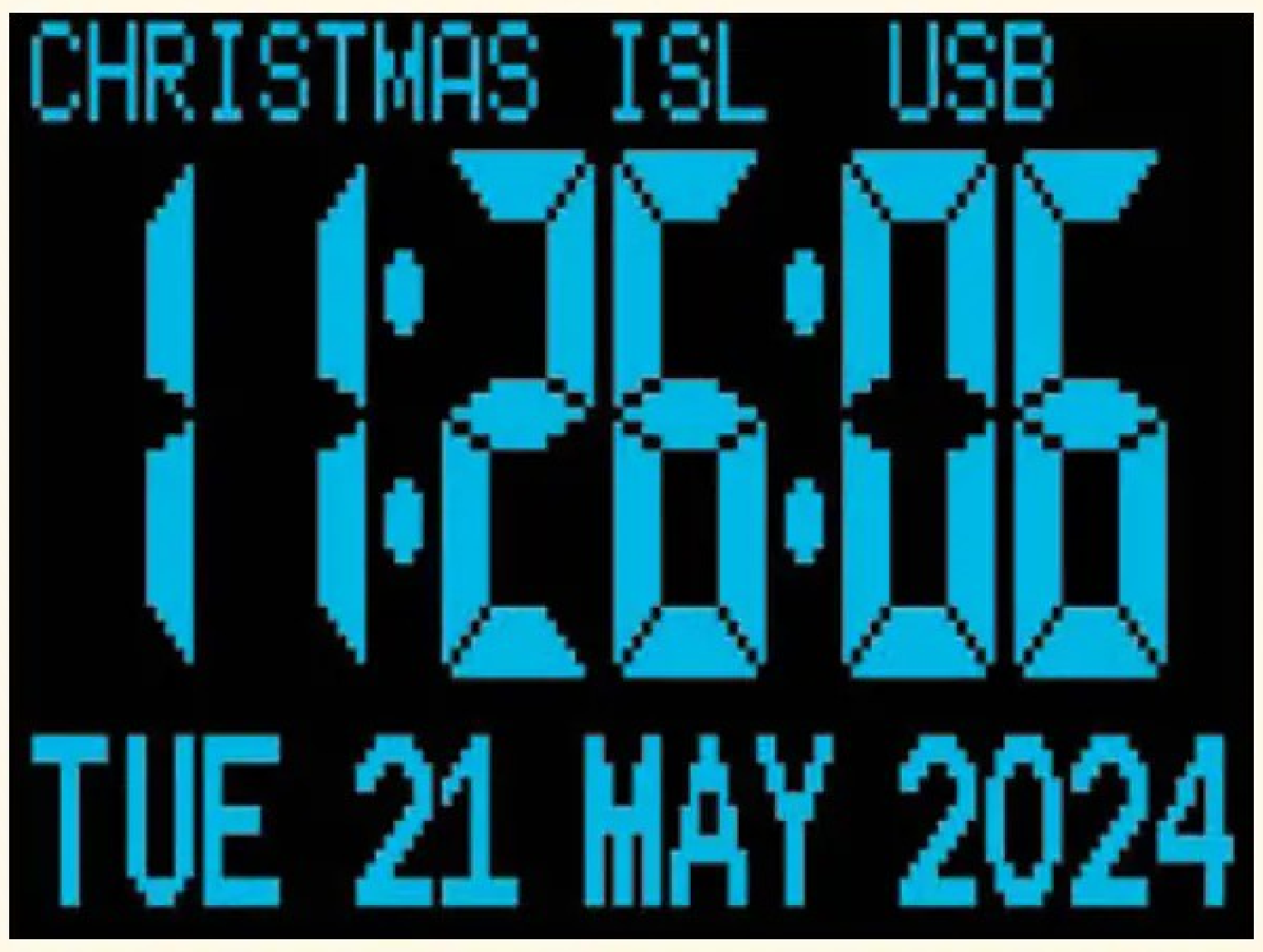
CUST TZ
OK TO SELECT
UP/DOWN TO EDIT
HOME TIMEZONE
SYDNEY
UTC+10:00 DST

Screen 7: the alarm clock is always based on the HOME timezone, which can be set here. Pressing OK also allows you to set the parameters for a custom time zone, including the default offset and when daylight saving starts and ends. This defaults to Greenwich Mean Time (GMT).



EXIT
GPS: ▲TURN ON
PRESS OK TO
EXIT SETTINGS

Screen 8: the last SETTINGS screen lets you return to regular operation and manually power the time source on and off with the UP and DOWN buttons. This is handy if you ever need to change the settings on the time source or update its firmware. It switches off when you exit SETTINGS.



CHRISTMAS ISL USB
11:26:06
TUE 21 MAY 2024

Screen 9: this shows an alternative font. The available time zones can be viewed by pressing the UP and DOWN buttons while the clock is showing. Pressing OK toggles between a 12-hour (AM and PM) or 24-hour clock. AM is shown by the letter A, PM by P and 24-hour mode with no letter.

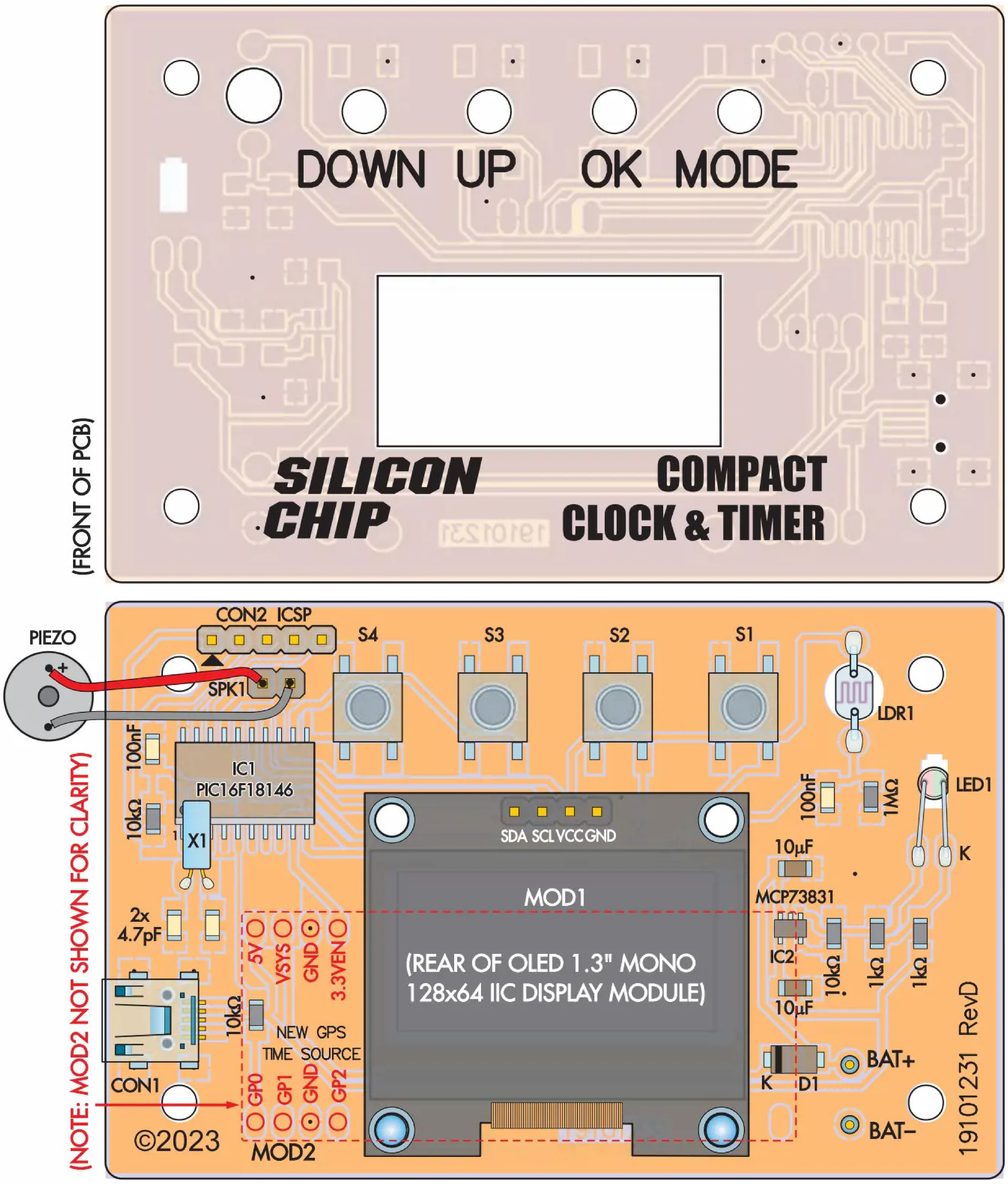

(FRONT OF PCB)
DOWN
UP
OK
MODE
SILICON CHIP
COMPACT CLOCK & TIMER
PIEZO
CON2 ICSP
SPK1
S4
S3
S2
S1
LDR1
LED1
K
100nF
IC1
PIC16F18146
10kΩ
X1
100nF
1MΩ
SDA SCL VCC GND
10µF
MOD1
MCP73831
IC2
10kΩ
1kΩ
1kΩ
10µF
(REAR OF OLED 1.3" MONO 128x64 IIC DISPLAY MODULE)
2x 4.7pF
5V
VSYS
GND
3.3VEN
NEW GPS TIME SOURCE
10kΩ
GP0
GP1
GND
GP2
CON1
K
D1
BAT+
BAT–
©2023
MOD2
19101231 RevD
(NOTE: MOD2 NOT SHOWN FOR CLARITY)

Fig.2: the PCB is populated mainly with surface-mounting components, plus a handful of through-hole parts fitted in surface-mounting fashion. This figure is shown at 140% of actual size for clarity.

OK if you have some experience), flux paste, solder-wicking braid, tweezers, a magnifier and a good light source.

You should have some sort of fume extraction gear; a fan close to your workspace pointing out an open window may be sufficient. You could also work outside or right next to an open window, which might also help with illumination.

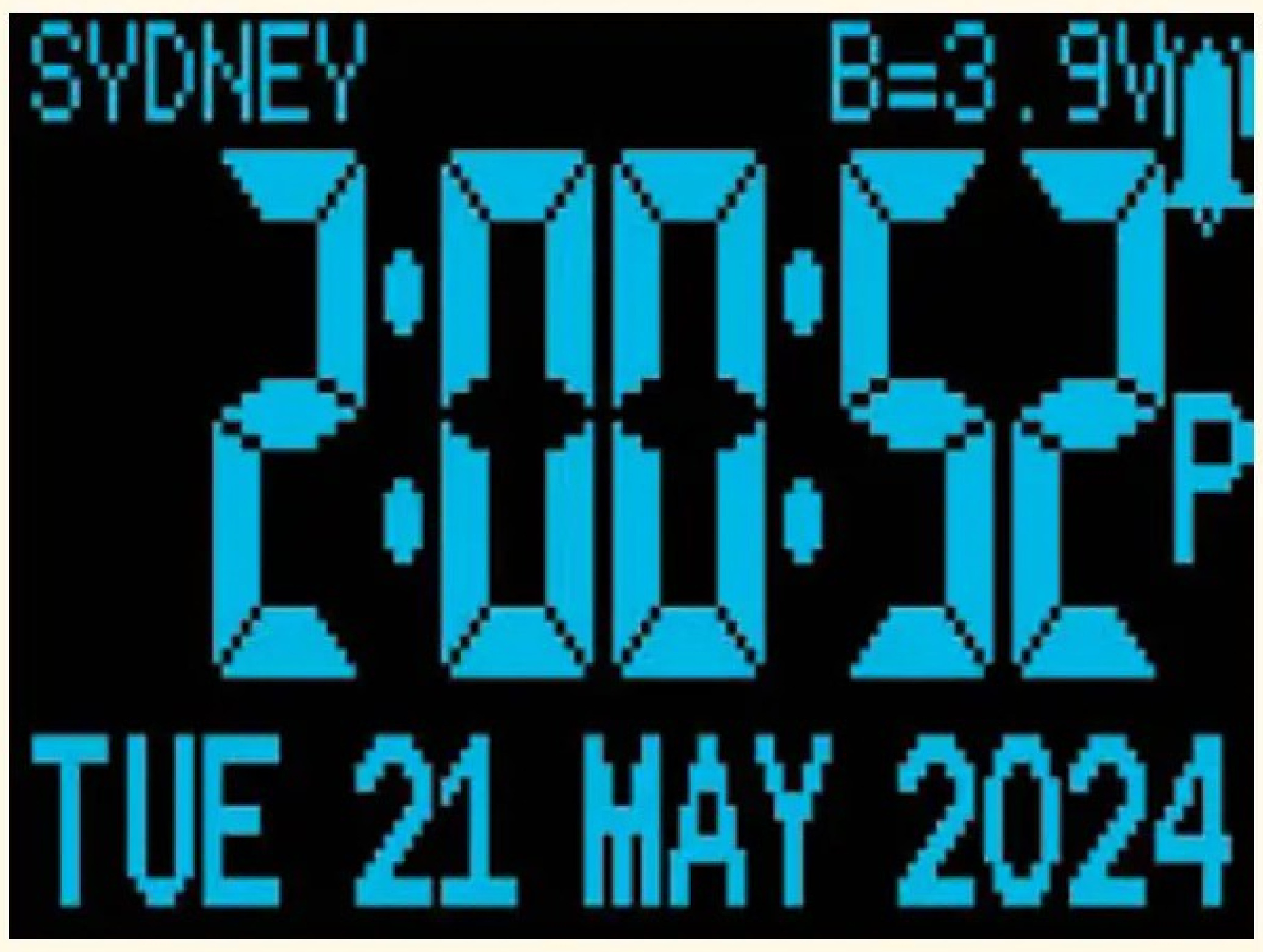
Note that some through-hole components are fitted in a surface-mounting fashion. You can get an idea of how these are installed by examining Fig.2, the PCB component overlay diagram, and the photos of the partially and fully populated PCB.

Start by soldering IC1 and IC2. IC1 must have its pin 1 marker aligned with that on the silkscreen, while IC2 will only fit one way as it has two pins on one edge and three on the other. Apply flux to the PCB pads and rest the chips in place.

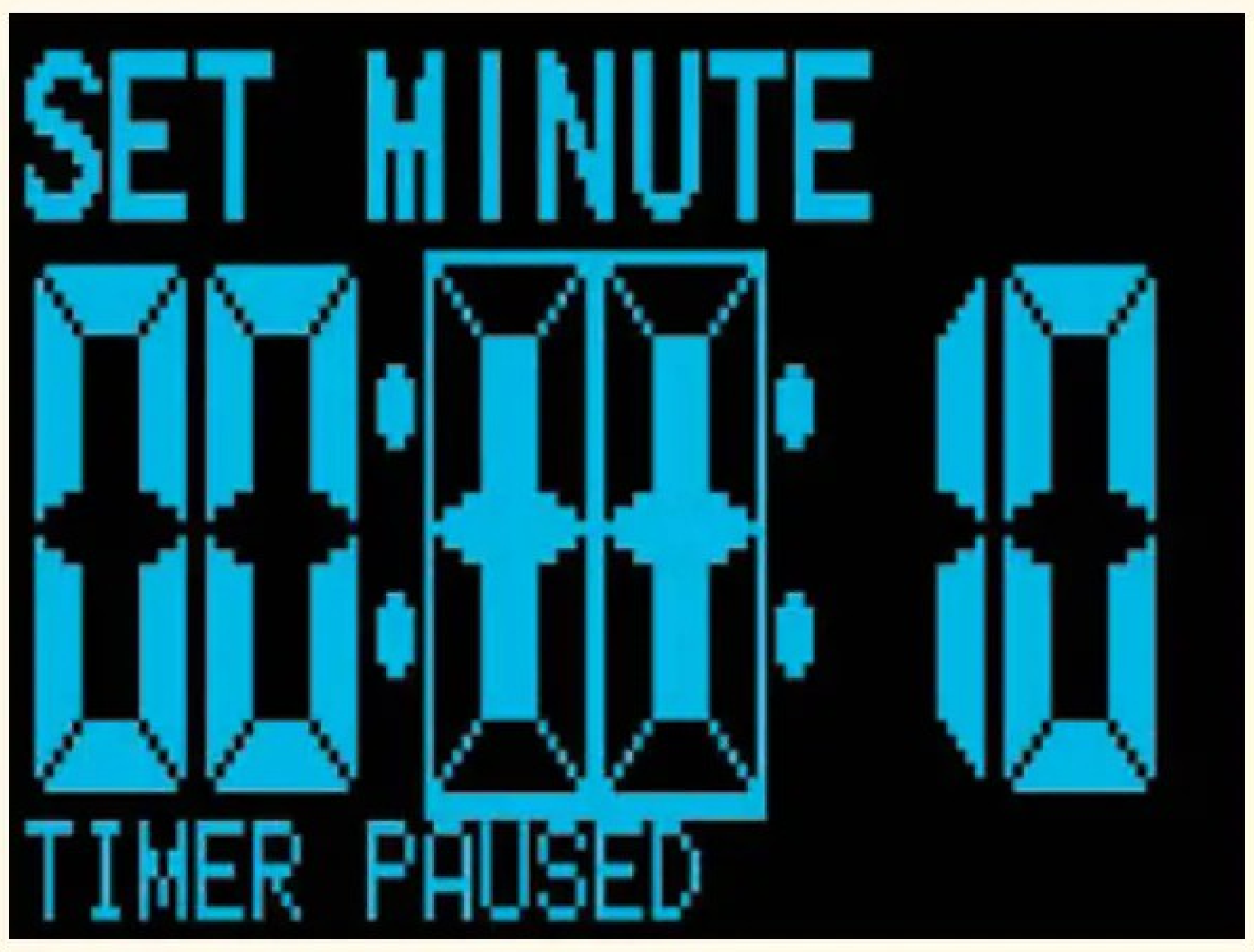
Tack one lead on each and check that the pins are aligned with the pads before soldering the others. If solder bridges form across any pin pairs, apply more flux and use the braid to draw out the excess.

Fit CON1 next. It has plastic locating lugs on its underside, making it easy to position. Solder the smaller pins and confirm that the part is flat against the PCB, then secure the larger pins with a generous amount of solder to ensure that the connector is firmly attached.

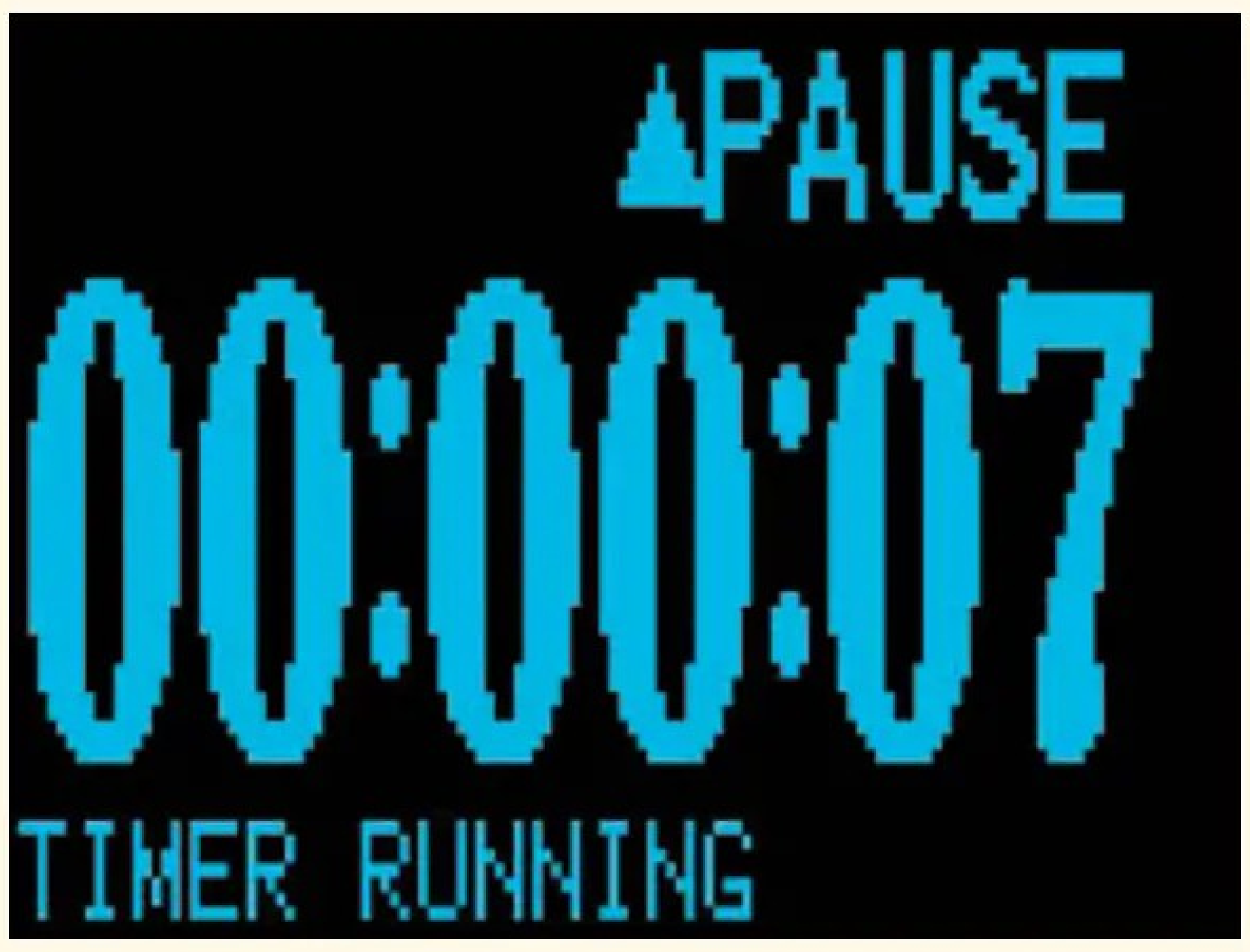
There are three different capacitor values (two of each), so do not mix them up, as they will not be marked with their values. Like the



Screen 10: the alarm symbol in the upper-right corner flashes while the alarm is sounding. Pressing OK stops the alarm. The top of the screen shows the battery status (voltage) display if USB power is not available. During a WiFi time source update, this will show "GPS".



Screen 11: pressing MODE switches to the Countdown Timer; you can then press OK until the SET screens appear. The UP and DOWN buttons on these screens change the clock's hours, minutes and seconds. The TIMER PAUSED status is shown when the timer is ready to start counting down.



Screen 12: pressing OK after setting the countdown time returns to the main Timer screen. Pressing UP will start (or resume) the Timer or pause it if it is running. DOWN will reset the Timer if it is paused or has expired. This screen shows the third font that's available (refer to Screen 4).

other parts, use some flux and tack one lead in place.

Confirm that the position is correct and that the first joint has solidified before soldering the other lead. Refresh the first lead if necessary (eg, with a touch of flux paste).

Follow by fitting the resistors similarly, then move on to D1, the schottky diode. Ensure that its cathode stripe is towards the K marking before soldering it. If this diode is reversed, power from the USB socket could feed directly into the battery, which would be catastrophic!

Next, mount the three through-hole components. Keep the lead offcuts from these, as they can be used to mount the OLED module later. Look closely at the photos since they are all arranged in a specific way.

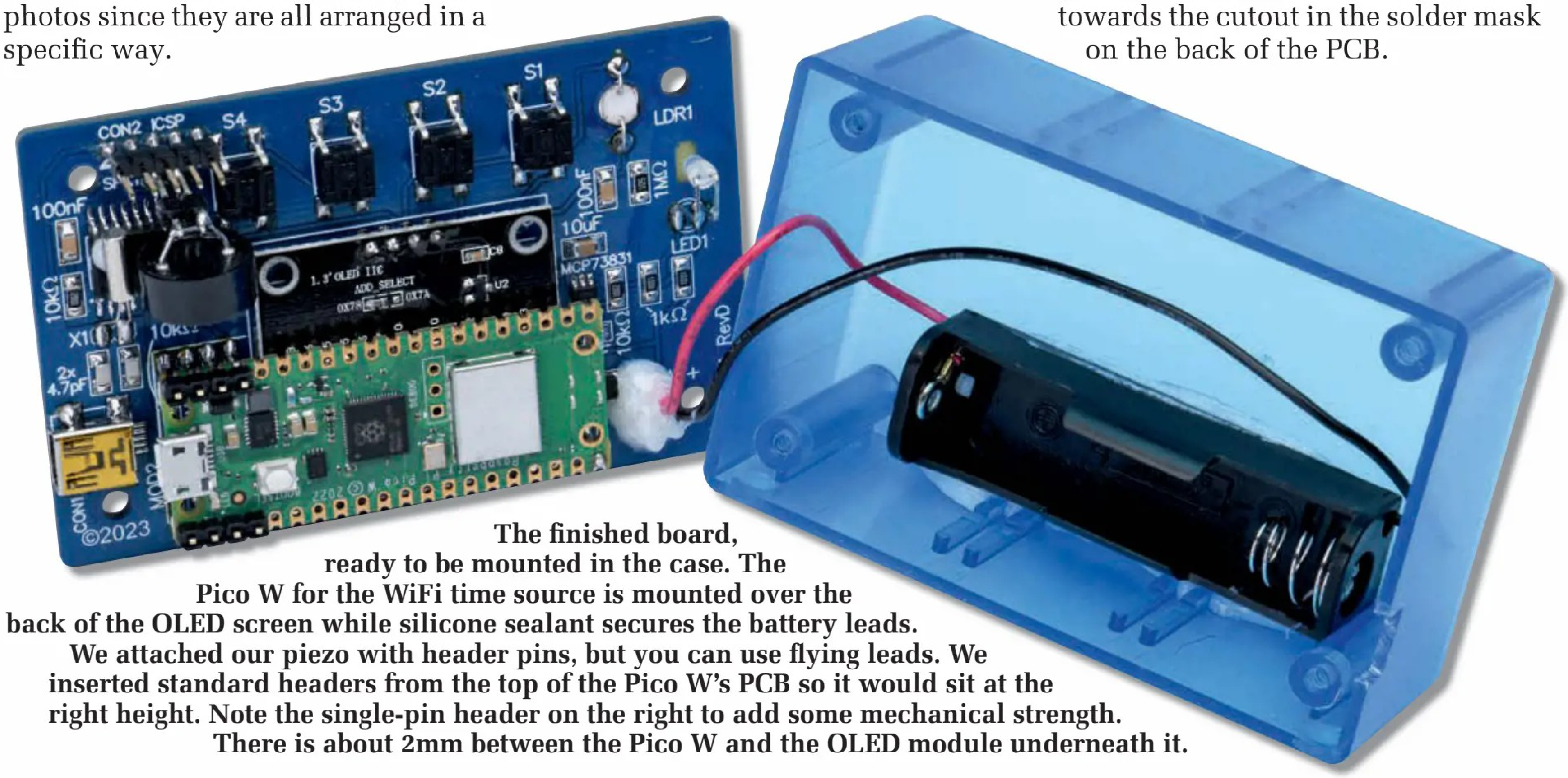
Crystal X1 is fitted so that it can be glued against IC1 later. It is not polarised, so it does not matter which lead goes to which pad. Splay the leads slightly to suit the pad spacing and bend them in an arc. They might also have to be trimmed. Once you have the leads adjusted, solder one to its pad, then tweak the leads if necessary before soldering the other lead.

For LDR1, trim one lead to around 5mm and bend it in a 180° arc. You can leave the other lead at its full length to ease handling. Press the LDR into the hole and tack the short lead in place. Adjust the position and orientation, if necessary, with the aim of having the front of the LDR flush with the outside of the PCB.

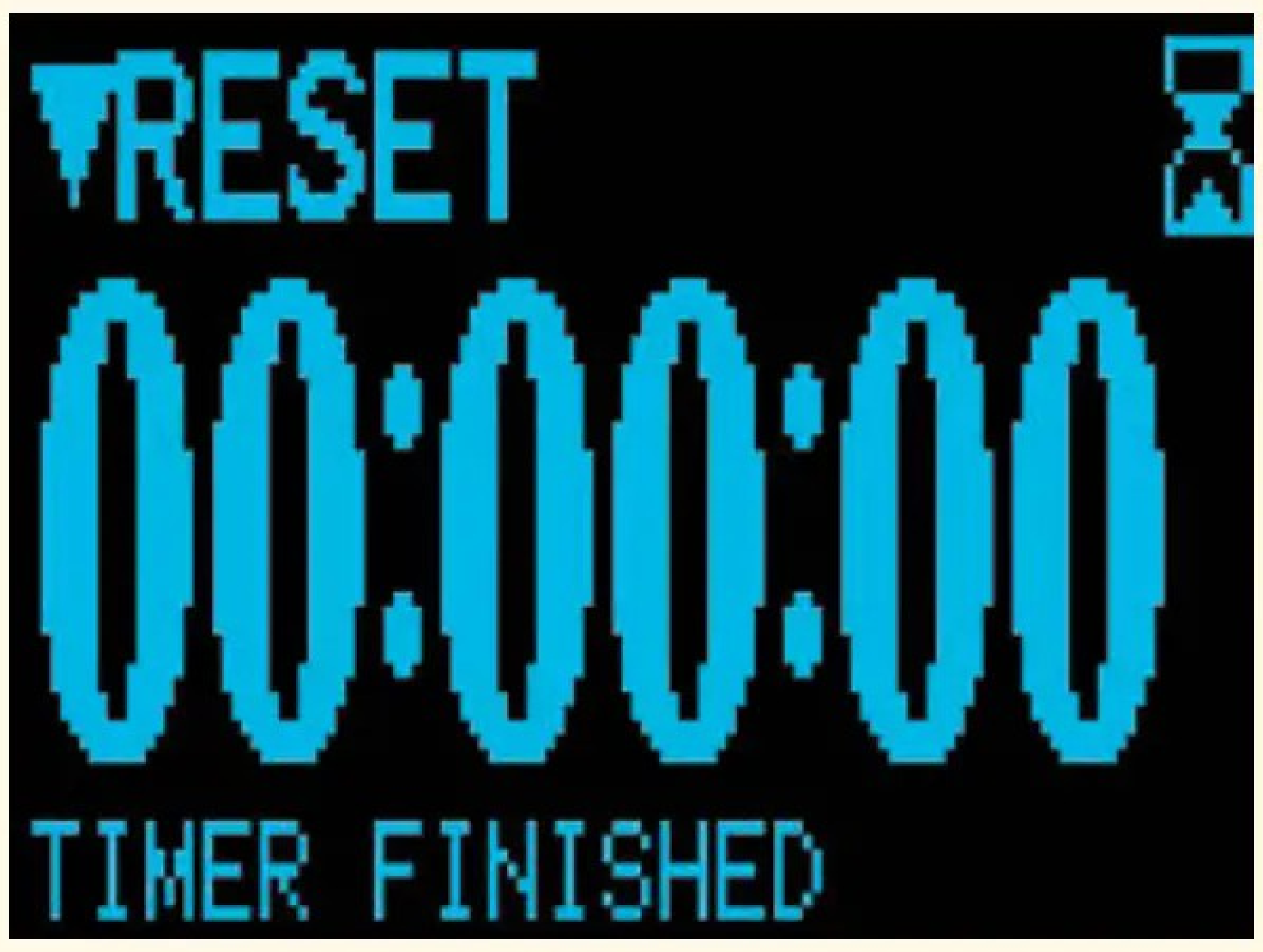
Then cut down the other lead and bend it into position over the other pad. Solder the second lead and refresh the first if necessary.

LED1 is a bit more tricky. The K cathode marking refers to the green LED of the bicolour device. So it's best to test the LED as some are marked (with the flat or longer lead) with reference to the red LED instead. Set a DMM on diode test mode and probe the leads. The red probe will indicate the anode of whichever colour LED lights up, and the black lead (cathode).

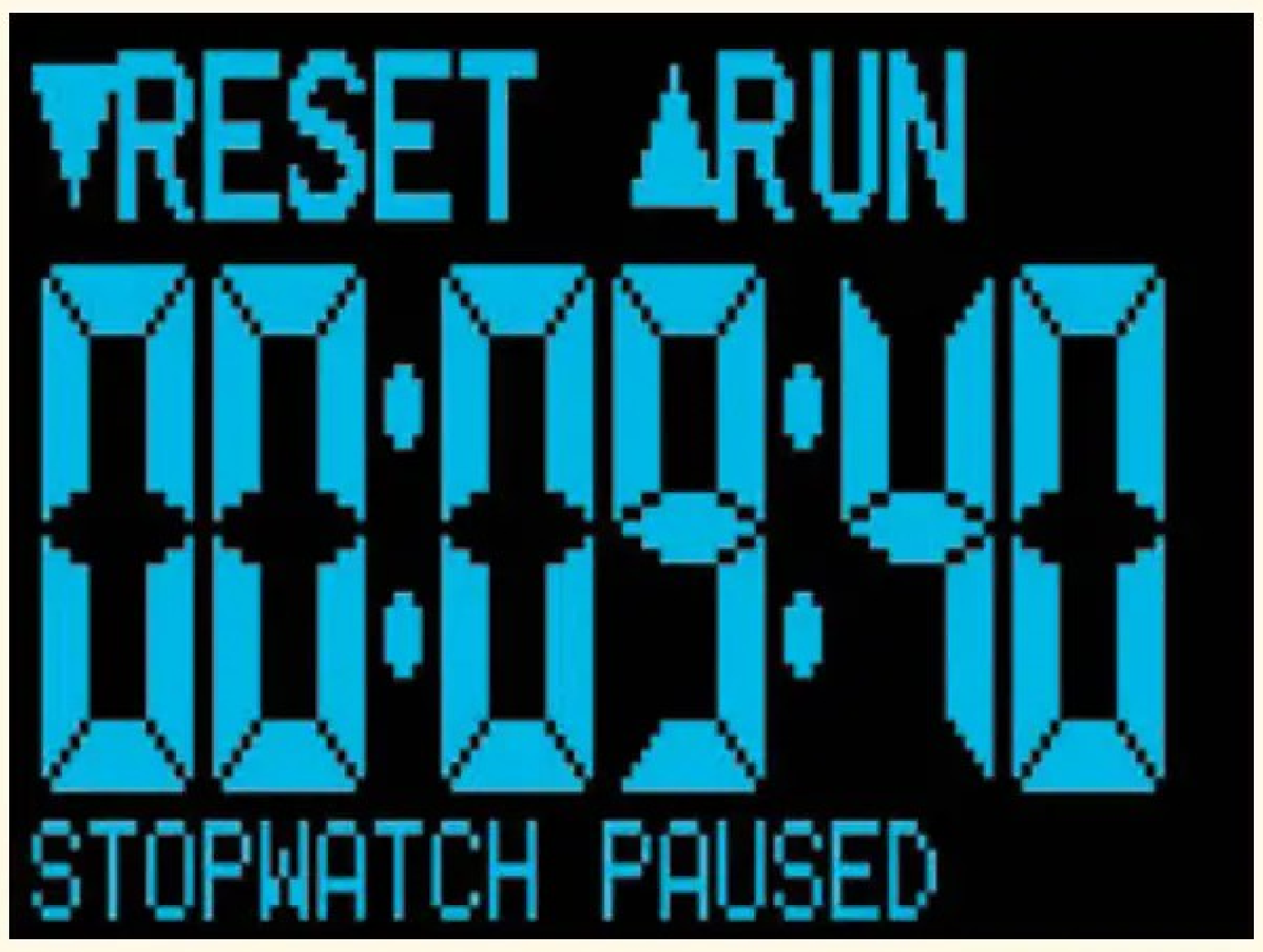
Bend the leads in the shape shown in the photos so that they reach the pads below. We've left quite a bit of lead on our prototype to make it easier to position and aim the LED so it shines towards the cutout in the solder mask on the back of the PCB.



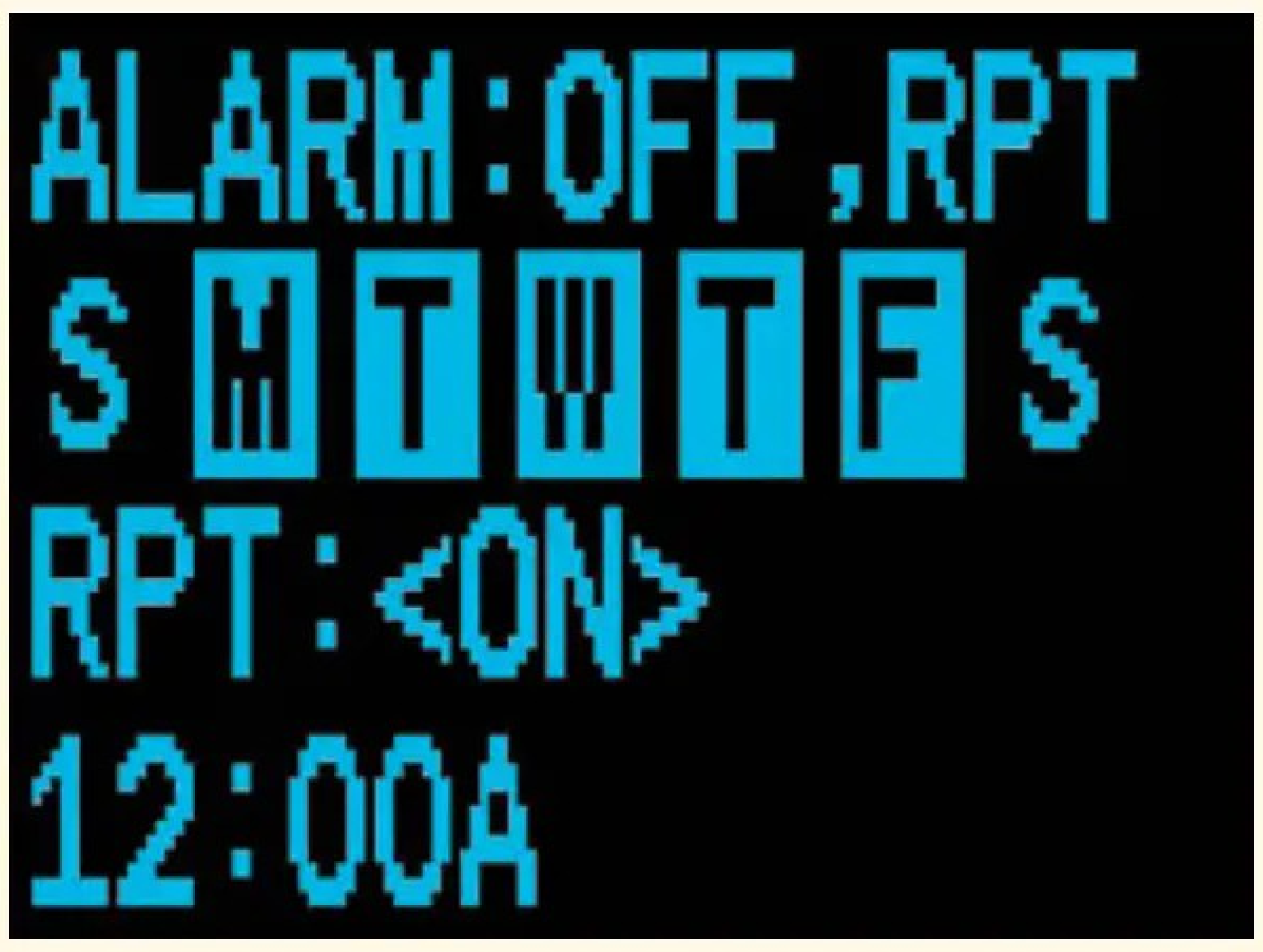
The finished board, ready to be mounted in the case. The Pico W for the WiFi time source is mounted over the back of the OLED screen while silicone sealant secures the battery leads. We attached our piezo with header pins, but you can use flying leads. We inserted standard headers from the top of the Pico W's PCB so it would sit at the right height. Note the single-pin header on the right to add some mechanical strength. There is about 2mm between the Pico W and the OLED module underneath it.


▼RESET ⧗
00:00:00
TIMER FINISHED

Screen 13: when the Timer finishes, you will see the hourglass symbol flashing in the corner of the display and hear the Timer tone. Press DOWN to stop the alert and reset the Timer. The Alarm and Timer icons and tones will occur in any operating mode except possibly SETTINGS.


▼RESET ▲RUN
00:09:40
STOPWATCH PAUSED

Screen 14: the Stopwatch is much simpler than the other modes. It is started, resumed or paused by pressing the UP button and can be reset while paused with the DOWN button. The timings are only updated every second by the timer interrupt.


ALARM:OFF,RPT
S M T W T F S
RPT:<ON>
12:00A

Screen 15: pressing MODE takes you to the Alarm clock setup. Press OK to cycle between the options, with UP increasing or enabling the setting and DOWN decreasing or disabling it. You can set the time to the nearest minute, choose days of the week, whether the alarm repeats and whether it is on.

Cleanup

Now is a good time to clean off any flux residue and closely inspect the board before proceeding to the next step. Use your flux's recommended solvent or some isopropyl alcohol to dissolve the flux and then allow the board to dry thoroughly.

Scrutinise the board with a magnifier to double-check that everything is soldered correctly and that there are no bridges. IC2 and CON1 have closely spaced pins, so look at them carefully.

Next, fit tactile switches S1-S4. The reverse mounting types are pretty nifty, but they will benefit from having their leads splayed back slightly to give the switch stems a bit more length projecting through the front of the PCB. Tack one lead on each switch in place and tweak the position so that they are centred in their holes.

It's worth spending some time getting this right, as it looks much better with the stems centred. It also eliminates the possibility of the stems binding. When you are happy, use a generous amount of solder to mechanically secure all four leads on each switch.

The next job is fitting OLED module MOD1. Attach a lead offcut to each of the four small PCB pads for MOD1, then thread the OLED over them, ensuring that the protective film is removed and the module is flat against the main PCB.

Solder the offcuts to the main PCB. The two large holes along the lower edge can be similarly attached to the large pads on the PCB below. Before doing this, ensure the OLED is square and symmetrical within the cutout.

At this stage, the assembly should look like the earlier partially completed PCB photo. The circuit is complete enough to do a basic test. If you still need to program IC1, do so before proceeding.

It is safe to apply power to the circuit via the programming header, CON2. Alternatively, you can apply power to the board by plugging a USB cable into CON1. The OLED should light up, and the LED will probably show both red and green because no battery is attached.

Check the voltage on the BAT+ terminal relative to BAT– (which is also circuit ground). It should be no more than 4.3V. If there are any problems, verify that diode D1 is correctly orientated.

The display will show a countdown from 60 seconds. If the countdown is not proceeding, there may be a problem with crystal X1.

Case cutting

The only necessary hole in the case is to allow the USB socket, CON1, to protrude out the side. Fig.3 shows the measurements, but this one is relatively easy to do by eye, especially if you use a transparent case like ours. Rest the PCB just inside the case with the USB socket against the wall of the case. You should be able to mark the outline of the socket using a pencil or similar.

Perform the downward cuts most of the way and then carefully flex the tab formed by the cuts. You can then use a sharp hobby knife to trim the hole to fit the USB socket comfortably.

Check that no parts prevent the PCB from sitting flush against the case. We've squeezed everything in tightly, but nothing should stop the case from closing.

If you have soldered a header to the CON2 ICSP pads, that could clash with the pillar inside the case. We found that trimming the plastic on the header was enough to prevent that, but you might consider removing the header if you only fitted it for programming IC1 initially.

Setting it up

If you haven't already done so, prepare the WiFi time source according to the instructions in the panel opposite. It's possible to program a Pico W in place or even modify its settings, but this is done more easily before it is attached to the PCB.

Male header strips are used to solder

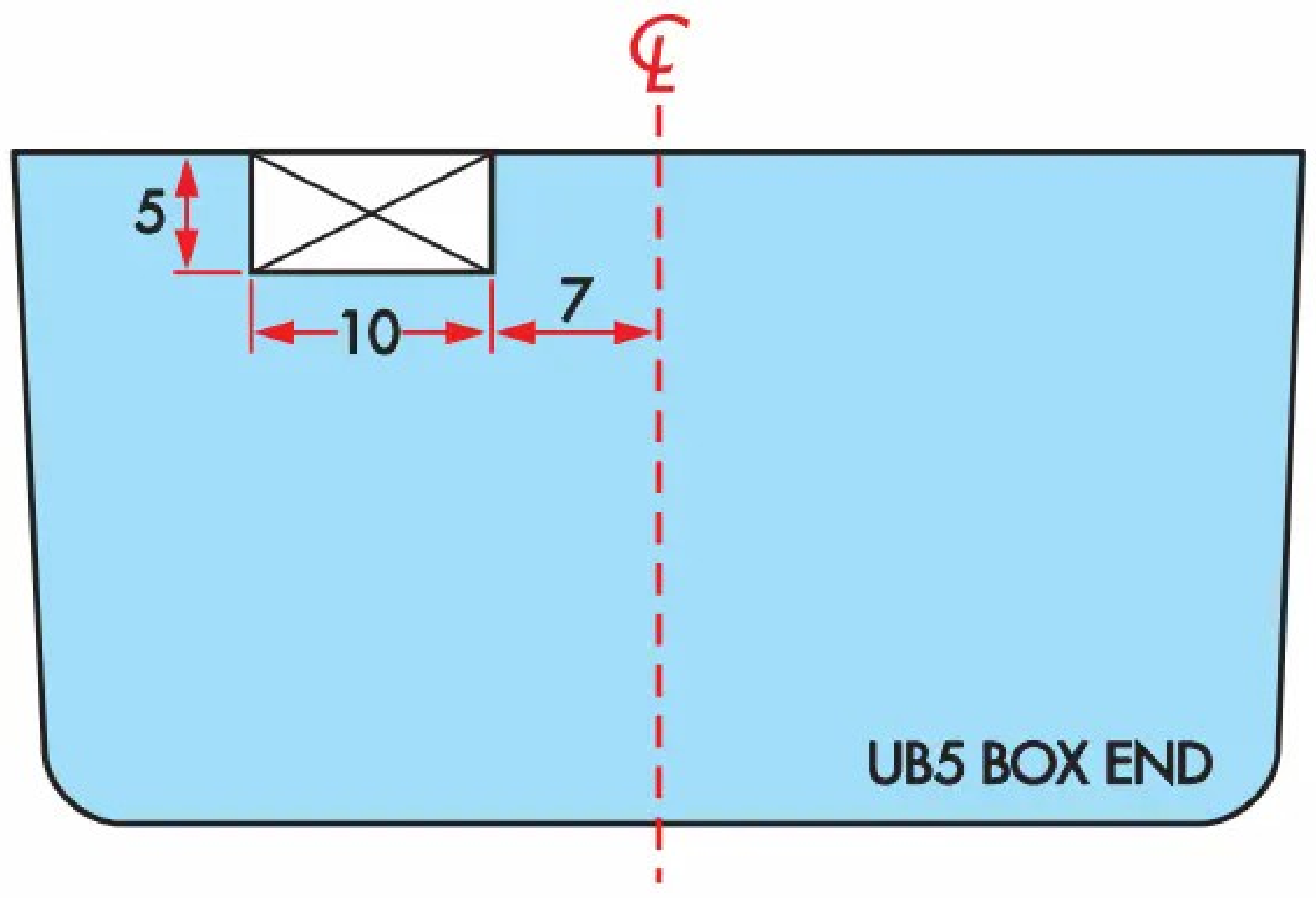


Fig.3: it is easy to cut out the small rectangular region for the USB socket by eye, allowing you to make it a snug fit. Here are the suggested dimensions of the cut if you wish to measure it out first (viewed from outside the box). All dimensions are in millimetres.

the Pico W to the PCB. Locating it behind the OLED module is the only way to get enough clearance to also fit the battery inside the enclosure. The bottom of the Pico W should be about 5mm above the PCB, leaving about a 2mm gap between the OLED module and the Pico W.

We achieved the correct height on our prototype by soldering the pin headers with the plastic shrouds above the Pico W's PCB. You can see the remnants of the shrouds in the photos (we trimmed off the tops of the pins).

Solder the two rows of four-pin headers to the USB end of the Pico W, keeping the pins square. Check that your positioning allows enough space to plug a USB cable into the Pico W; the cable's bezel should just clear the CON1 USB socket on the main PCB.

Solder the tips of one of the pin headers to the main PCB and check that everything is aligned. Next, solder the single-pin header from pin 20 of the Pico W to the main PCB. There is a pad for this adjacent to D1.

When everything looks correct, you can proceed to add a fillet of solder from each of the Pico W pins back to the main PCB, securing it. Trim any excess height from the pins to give the battery as much clearance as possible.

Solder the battery holder to its terminals marked BAT+ and BAT–, taking great care that the polarity is correct. The way we installed the battery holder in the case allowed us to shorten the red (BAT+) wire.

Also solder the piezo element to the SPK1 pads. We used header pins, but you could use flying leads (such as offcuts from the battery leads) to allow the piezo to be glued to the case. In that case, we also recommend drilling a hole in the case to enable the sound to escape.

Now glue the battery holder into the case as shown in the photos. Also apply neutral-cure silicone sealant to the BAT+ and BAT– terminals to insulate the pads and secure the wires mechanically.

If you have the piezo on flying leads, glue it to the case now. You can also add a dab of glue to the crystal to secure it to the top of IC1. After that, wait for all the adhesive to cure fully.

Now insert the battery into the holder. The screen should light up, and you should see the LED on the

Setting up the WiFi time source

The June 2024 project article for the Wi-Fi Time Source for GPS Clocks details how the time source works, but this overview should have enough information for you to set it up.

You will need a Raspberry Pi Pico W microcontroller board programmed with the time source firmware, which can be downloaded from **siliconchip.au/Shop/6/188**

Hold the white BOOTSEL button of the Pico W while connecting it to a computer. This will put it into bootloader mode, and you should see a drive named "RPI-RP2" appear. Copy the "NEW_CLAYTONS_1.uf2" file to that drive to upload the firmware.

If all is well, the LED on the Pico W should light up, the drive should disappear, and you will have a virtual USB serial port available. Use a serial terminal program like TeraTerm on Windows to connect to the port (you could use minicom on Linux).

Set the terminal to use CR or CR+LF as the line ending and press Enter. It should then show the status and command menu. The following is not a comprehensive overview of the time source's capabilities, but it will be sufficient to program it for use with the Clock/Timer.

Use command 9 (press the 9 key followed by Enter) and then enter the two-letter country code (eg AU, NZ, US, UK etc). If you are likely to use the Clock internationally, the global "XX" setting is safest.

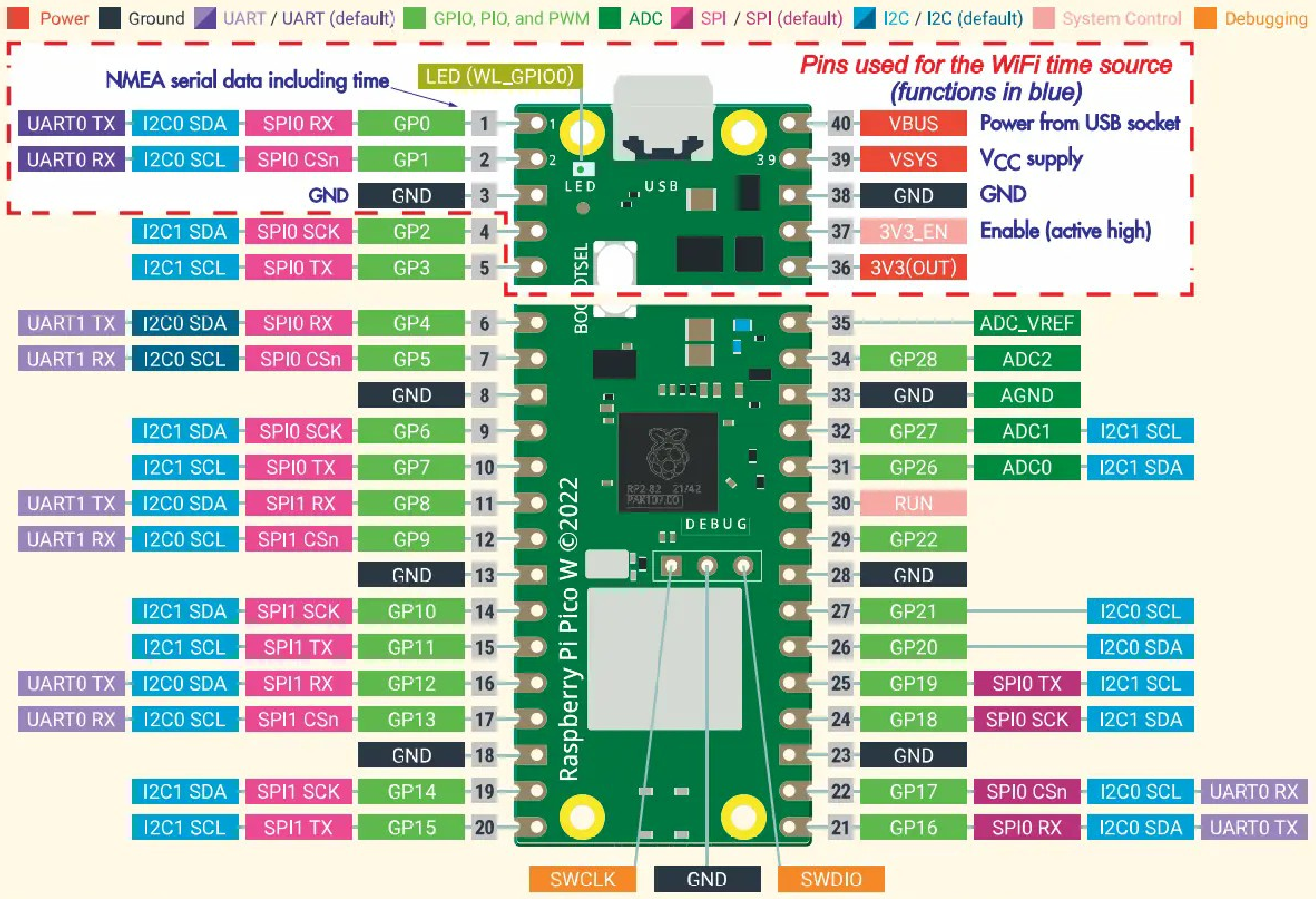
Next, use command 8 (8, Enter) to save that setting to flash and follow with command J (capital J, Enter) to reboot the time source. This ensures the WiFi radio is initialised with the correct country code at power-up.

Use command 1 to run a scan of WiFi networks. The nearby networks should be listed with a number next to each one. Then run command 4 with one of the listed numbers as a parameter. For example, if your home WiFi network is listed first, as number 0, type 40, then Enter.

You will then be prompted for the password; type it, then press Enter again. Use command 7 to test the network and, if all is well, use command 8 to save the settings to flash memory. Use J to reboot again and check that the time source connects to the network.

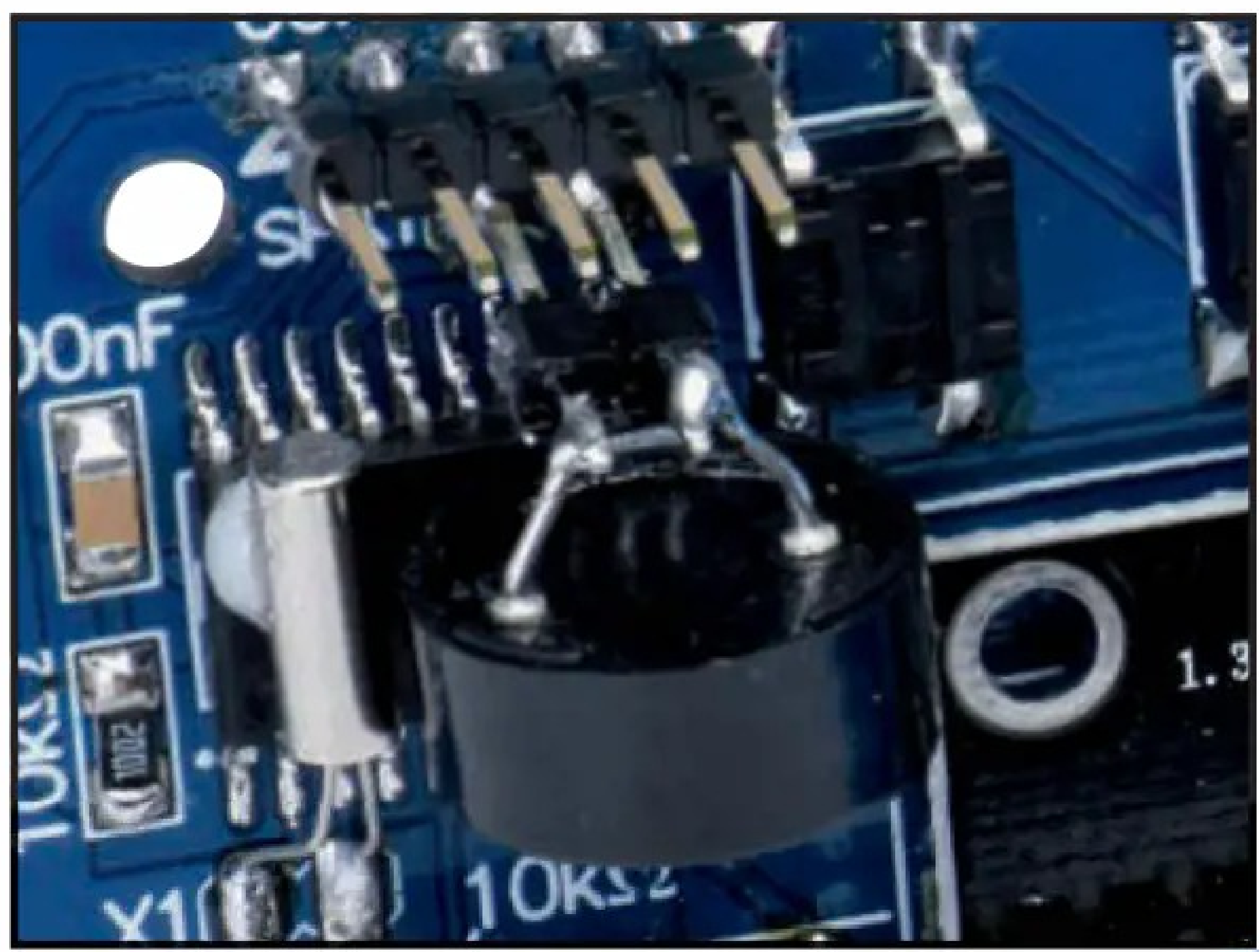
The LED should change from solid to flashing when it successfully connects to a network. Flashes occur in groups of three if everything is working and the time has been acquired from the NTP service. You can add multiple networks by running commands 1, 4, 7 and 8 when in the vicinity of each network.

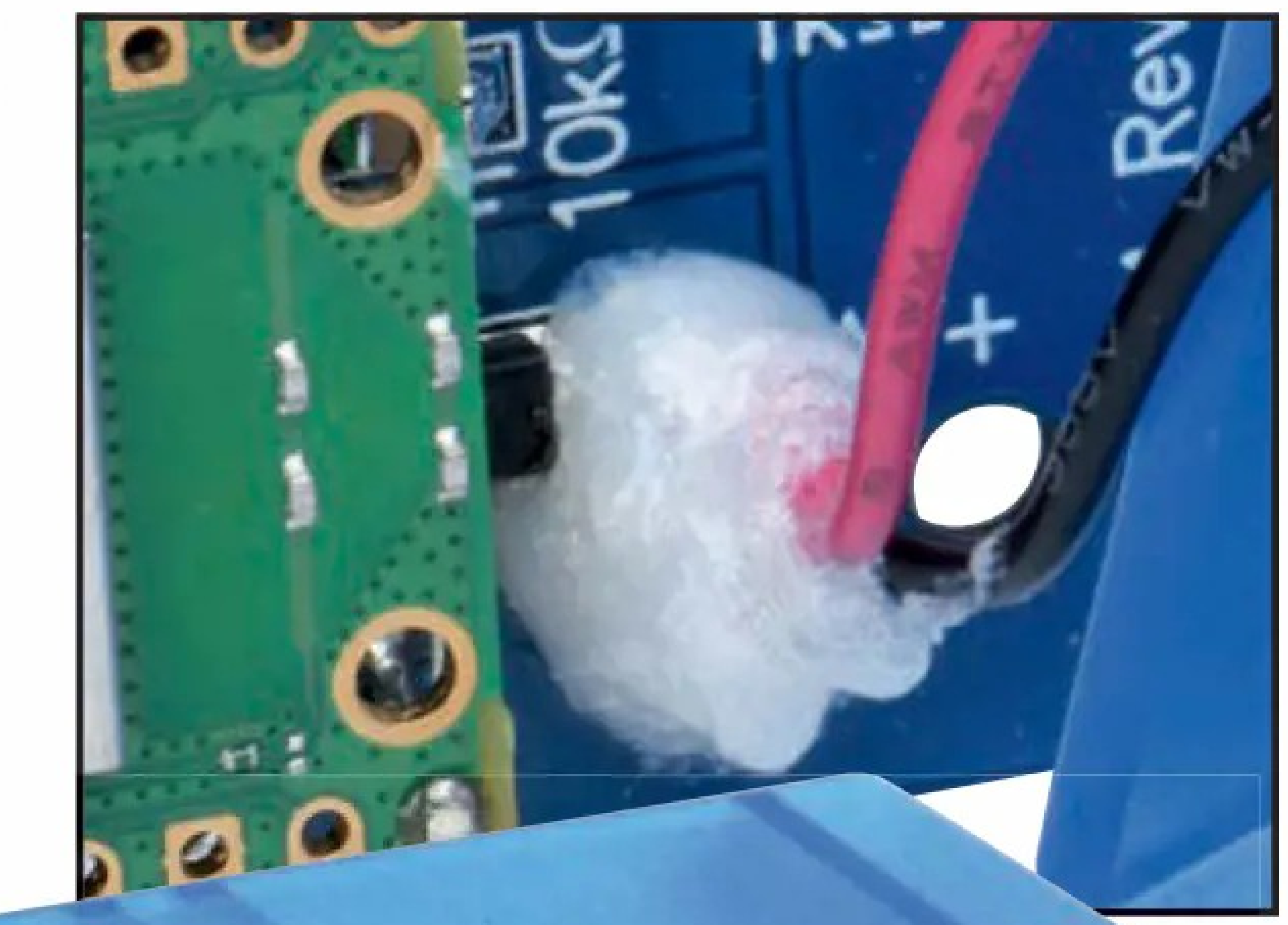
If you see groups of three flashes, the time source is working as expected. If you run into problems, you can also examine the output and debugging data to determine the source of the problem. Many other settings are available, but there is little need to change any of them. The Compact Timer has been designed to work with the WiFi time source's default configuration.

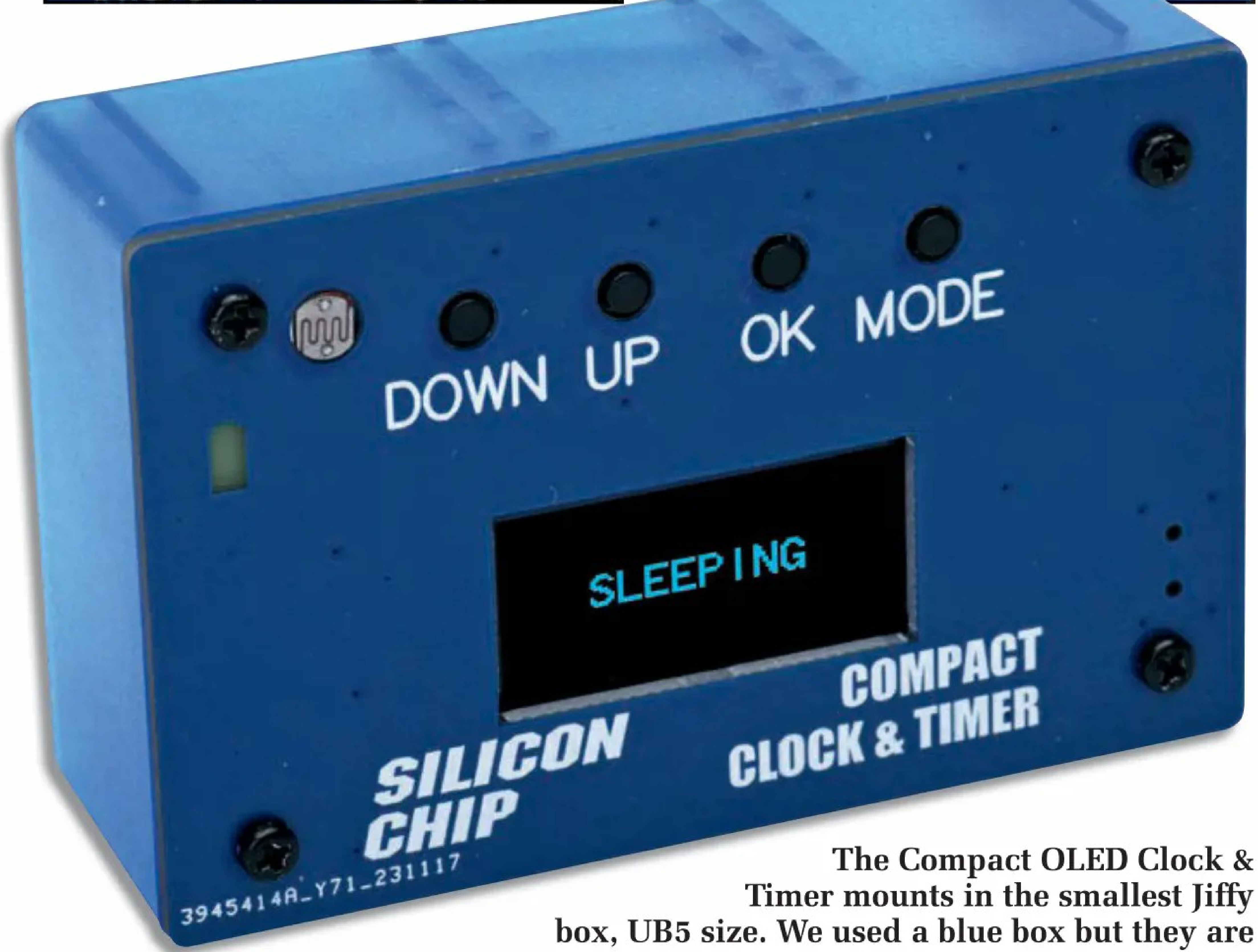


Power
Ground
UART / UART (default)
GPIO, PIO, and PWM
ADC
SPI / SPI (default)
I2C / I2C (default)
System Control
Debugging
Pins used for the WiFi time source
(functions in blue)
NMEA serial data including time
LED (WL_GPIO0)
UART0 TX I2C0 SDA SPI0 RX GP0 1
UART0 RX I2C0 SCL SPI0 CSn GP1 2
GND GND 3
I2C1 SDA SPI0 SCK GP2 4
I2C1 SCL SPI0 TX GP3 5
UART1 TX I2C0 SDA SPI0 RX GP4 6
UART1 RX I2C0 SCL SPI0 CSn GP5 7
GND 8
I2C1 SDA SPI0 SCK GP6 9
I2C1 SCL SPI0 TX GP7 10
UART1 TX I2C0 SDA SPI1 RX GP8 11
UART1 RX I2C0 SCL SPI1 CSn GP9 12
GND 13
I2C1 SDA SPI1 SCK GP10 14
I2C1 SCL SPI1 TX GP11 15
UART0 TX I2C0 SDA SPI1 RX GP12 16
UART0 RX I2C0 SCL SPI1 CSn GP13 17
GND 18
I2C1 SDA SPI1 SCK GP14 19
I2C1 SCL SPI1 TX GP15 20
40 VBUS Power from USB socket
39 VSYS VCC supply
38 GND GND
37 3V3_EN Enable (active high)
36 3V3(OUT)
35 ADC_VREF
34 GP28 ADC2
33 GND AGND
32 GP27 ADC1 I2C1 SCL
31 GP26 ADC0 I2C1 SDA
30 RUN
29 GP22
28 GND
27 GP21 I2C0 SCL
26 GP20 I2C0 SDA
25 GP19 SPI0 TX I2C1 SCL
24 GP18 SPI0 SCK I2C1 SDA
23 GND
22 GP17 SPI0 CSn I2C0 SCL UART0 RX
21 GP16 SPI0 RX I2C0 SDA UART0 TX
LED
USB
BOOTSEL
DEBUG
Raspberry Pi Pico W ©2022
SWCLK
GND
SWDIO

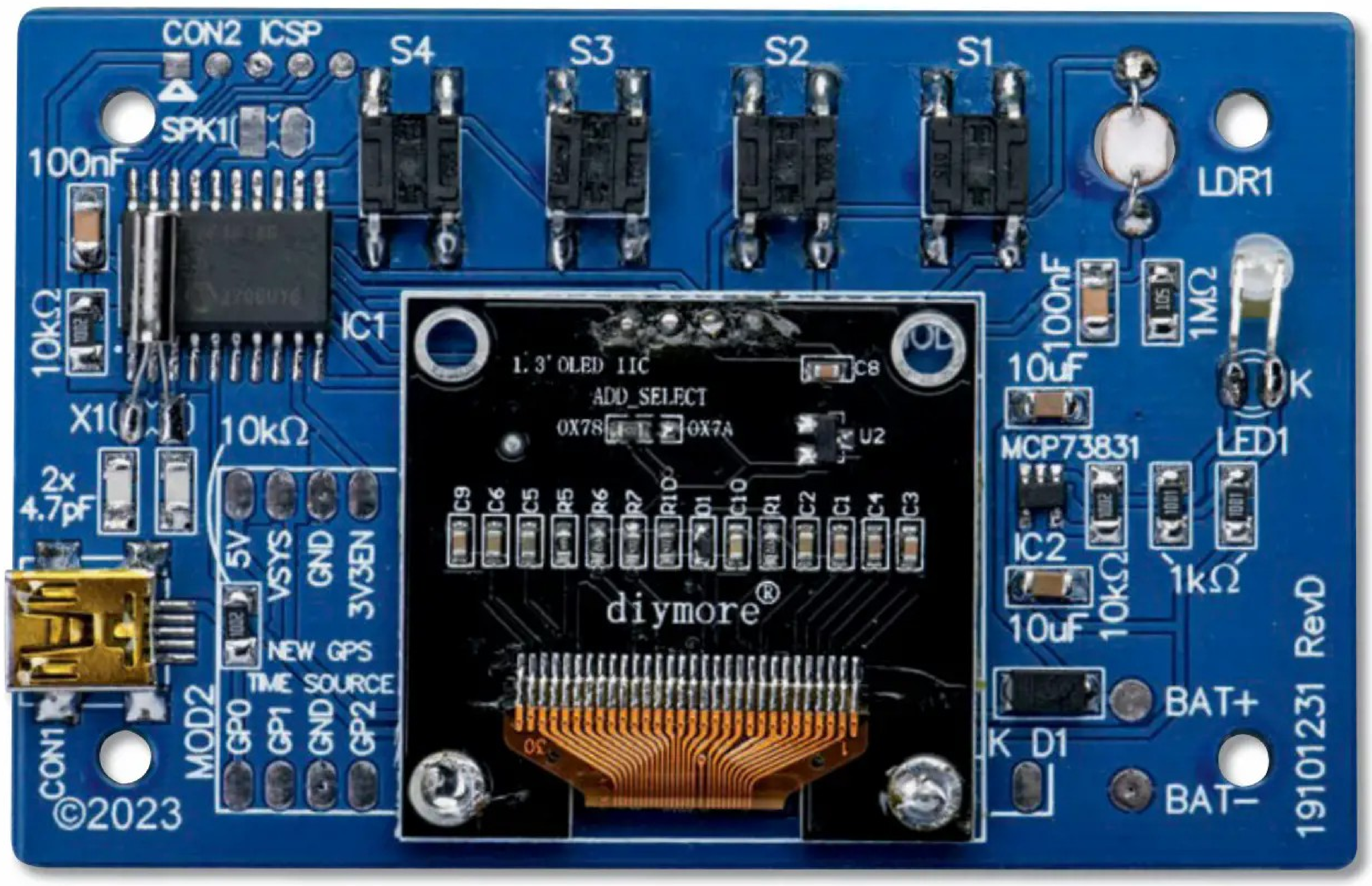
With the important pins at one end of the Pico W, near the USB connector, it's easy to connect to the Clock/Timer PCB without using up much space. Pins 1, 3, 37, 38, 39 and 40 are used in the circuit, while pins 2, 4 and 20 are also connected to add mechanical stability.






DOWN
UP
OK
MODE
SLEEPING
COMPACT
CLOCK & TIMER
SILICON
CHIP

The Compact OLED Clock & Timer mounts in the smallest Jiffy box, UB5 size. We used a blue box but they are hard to get. The controls are simple and, once configured, it will always keep time to within a few seconds. The Clock/Timer is shown in its lowest power mode – use the MODE button to switch to the Clock display, then hold the OK button until this screen appears. It will wake when the OK button is pressed again, if an alarm occurs or the countdown timer expires.


CON2
ICSP
S4
S3
S2
S1
SPK1
100nF
LDR1
IC1
10kΩ
X1
10kΩ
2x 4.7pF
1.3' OLED IIC
ADD_SELECT
diymore
100nF
1MΩ
10uF
MCP73831
LED1
K
IC2
10kΩ
1kΩ
10uF
5V
VSYS
GND
3V3EN
NEW GPS
TIME SOURCE
GP0
GP1
GND
GP2
MOD2
CON1
©2023
BAT+
BAT–
D1
K
19101231 RevD

The LED and LDR are standard through-hole parts that have been surface-mounted to avoid solder joints on the front of the PCB (see Fig.2). We have also splayed out the leads of the switches to bring them closer to the PCB.

Pico W come on after a second. Carefully fit the PCB into the case, being careful not to pinch any wires.

Attach the rubber feet to the bottom edge of the box to complete the assembly. Now is a good time to plug in a USB cable to charge the battery fully.

Setup and usage

The Clock/Timer will attempt to set the time via NTP when powered on, so allow that to happen. We've included several screenshots of the Clock and Timer in various states. Refer to those screen captions for the basics of setting up and using it, in the order shown.

The low power mode (with the screen off) can be activated by holding the OK button in the Clock Mode. When the SLEEPING message appears, release the button. Pressing OK again will reactivate the display.

The alarm and timer will also reactivate the display when they sound their respective alert tones. If both alerts are active, their tones and icons will alternate.

The software is set to perform several actions at five minutes past the hour (relative to UTC). This is when the clock trimming will occur if you wish to observe it. The automatic time updates occur at five minutes past UTC midnight. That will be, for example, 10:05am in Sydney or 11:05am during daylight saving time.

The crystal trimming routine needs two synchronisations before it will make adjustments, so you might have to wait a day or two before the trimming settles. Once that has occurred, the clock should always be within two seconds of the correct time.

Operation of the LDR and OLED brightness is fully automatic. Small adjustments are made so that the changing brightness is not noticeable; it can take up to a minute to settle after a change in ambient lighting. If you find the OLED is too bright, try decreasing the value of the 1MΩ resistor in series with the LDR.

Summary

The Compact OLED Clock and Timer is a portable and easy-to-use device that boasts features that even some clock apps do not. Once set up, it will maintain time within a few seconds as long as it can connect to a WiFi network daily. PE

Subscribe to SILICON CHIP

SILICON CHIP
RGB LED ANALOG CLOCK
RNBD451 Bluetooth Module
VERSATILE BATTERY CHECKER
ElectroneX
Melbourne Exhibition Centre 7 – 8 May 2025

SILICON CHIP is one of the best DIY electronics magazines in the world. Each month is filled with a variety of projects that you can build yourself, along with features on a wide range of topics from in-depth electronics articles to general tech overviews.

	Print	**Combined**	**Online**
6 months	$100 ≈£50 ≈€58	$110 ≈£55 ≈€64	$50 ≈£25 ≈€29
1 year	$195 ≈£97.50 ≈€113	$215 ≈£112.50 ≈€125	$95 ≈£47.50 ≈€55
2 years	$380 ≈£190 ≈€220	$415 ≈£207.50 ≈€241	$185 ≈£97.50 ≈€107

- *Combined subscriptions include both the printed magazine and online access.*
- *Prices are in Australian dollars (AUD) unless otherwise specified.*
- *Prices listed are for everywhere in the world except Australia and New Zealand. View siliconchip.com.au/Shop/SubRates for a full list of current prices.*
- *Price estimates are just that and should only be used as a guide. We can only take payment in the form of Australian dollars.*

Try our Online Subscription – now with PDF downloads!

An online issue is perfect for those who don't want too much clutter around the house and is the same price worldwide. Issues can be viewed online, or downloaded as a PDF.

Published in SILICON CHIP

RGB LED 'Analog' Clock; May 2025

Versatile Battery Checker; May 2025

Pico/2/Computer; April 2025

To start your subscription go to siliconchip.com.au/Shop/Subscribe

A human in the loop

Techno Talk

Max the Magnificent

HITL used to stand for hardware-in-the-loop (eg, hardware as part of a simulation). Now, HITL also stands for human-in-the-loop, where human judgement is added to automated systems.

I'm rarely exposed to something new in isolation. Usually, it doesn't take long before I'm up to my ears in more instances or variants of the same thing. Take robots, for example. In my previous column, we considered some of the latest and greatest developments in humanoid robots.

As we noted in that column, human environments—like homes, offices and factories—are designed for humans. As a result, robots with legs are better than their wheeled counterparts for navigating things like stairs. Similarly, robots with humanoid arms and hands can employ tools, operate machines and manipulate objects that were intended for use by people.

It's also true that when robots look more like people, they can interact with humans more naturally, using familiar mannerisms (gestures, movements and motions). This facilitates their use in roles like companions, caregivers, receptionists and tutors; cases where human-like interaction will dramatically enhance the user experience (or is intrinsic to the role).

First, the good news

Robots have been around for quite some time. The first successful industrial robot in the modern sense was arguably the Unimate from Unimation Inc, which was first deployed in a General Motors factory in 1961. The Unimate was an industrial robot arm, not a humanoid robot, but it marked the first step (no pun intended) toward real-world robotics as we think of it today.

Today's robots are a world away from the Unimate. Some very impressive examples of humanoid robots are:

- Apollo from Apptronik (https://youtu.be/pymvNott6nw)
- Atlas from Boston Dynamics (https://youtu.be/I44_zbEwz_w)
- Digit from Agility Robotics (https://youtu.be/0dexTZ7r02Q)
- Optimus from Tesla (https://youtu.be/jQd128YTtQQ)
- NEO Gamma from 1X (https://youtu.be/uVcBa6NXAbk)

Now for the bad news

These examples are on the cutting edge of humanoid robotics. They can walk, balance, carry objects, climb stairs and perform basic manual tasks (sometimes with surprising grace). However, they also share several attributes and limitations that currently make them more futuristic marvels than everyday tools.

For example, they rely on advanced, heavy-duty artificial intelligence (AI) for perception, decision-making, balance, navigation and manipulation tasks. These AIs are typically powered by artificial neural networks (ANNs), reinforcement learning and large-scale training datasets. In many cases, they depend on cloud-based assistance for complex tasks or learning.

Despite their robust appearance, these robots are sensitive to falls, environmental challenges and unpredictable variables. So regular maintenance, calibration and software updates are essential. Their battery life is often measured in hours, not days. Charging and power management are still major concerns.

Also, there are legal, ethical and safety concerns with robots interacting with humans in public or private spaces.

Then there's the cost. Estimates vary, but these robots can be in the hundreds of thousands of dollars range.

Even Tesla's Optimus, which might aim for a \$20–30k (£15-22.5k) price point, is still far from affordable by the average person.

Feeling our age

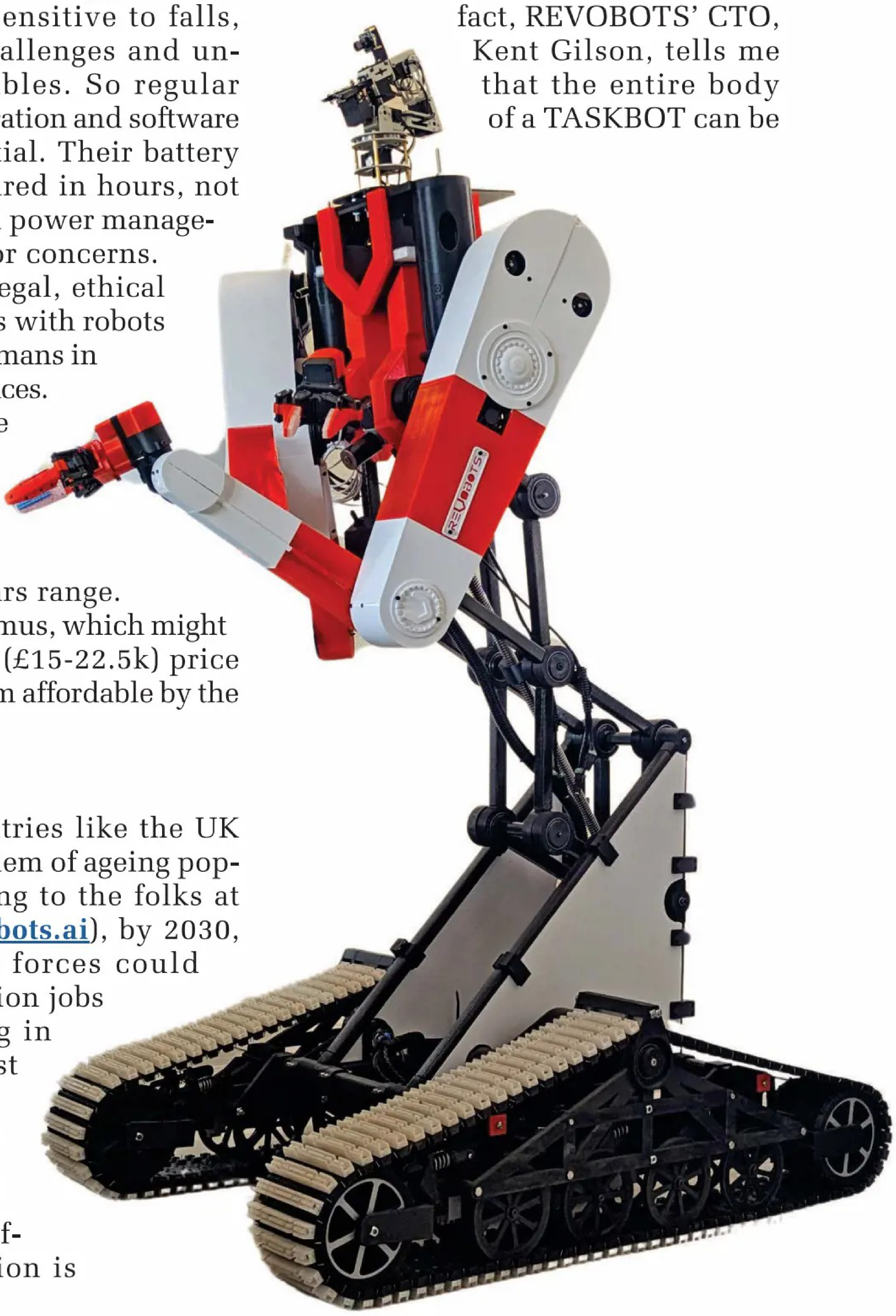
Developed countries like the UK are facing the problem of ageing populations. According to the folks at REVOBOTS (revobots.ai), by 2030, shrinking labour forces could leave over 85 million jobs unfilled, resulting in \$8.5 trillion in lost annual revenue if nothing is done. High inflation and stagnant wages will compound the difficulty; the situation is only likely to get worse for businesses looking for staff.

Perhaps this explains why Markets and Markets are predicting that the humanoid robot industry will exceed \$17 billion by 2027, growing at a compounded annual rate of more than 50% from 2022.

Meet TASKBOTs

All of this leads us to TASKBOTs from REVOBOTS. These beauties can be fixed or mobile with one or multiple arms, where the arms can handle a 150-pound (70kg) payload. Their end-effectors (hands) can have sub-micron accuracy.

One of the TASKBOT's claims to fame is that its structural components are 3D-printed. In fact, REVOBOTS' CTO, Kent Gilson, tells me that the entire body of a TASKBOT can be



produced using only around $1000 (£750) worth of raw materials. That sounds good to me!

TASKBOTs are equally parsimonious on the computational front. Most of today's humanoid robots are controlled by large language models (LLMs) that demand expensive, high-end, power-guzzling processors. By comparison, the TASKBOT employs a set of small language models (SLMs) that are combined hierarchically, can be trained quickly and have a very small memory footprint.

In fact, TASKBOT's computational requirements can be satisfied by a $50 dual-core ZYNQ SoC (system-on-a-chip) FPGA (field-programmable gate array) from Xilinx/AMD in each arm, plus an Arduino to control the track subsystem in the base.

A human in the loop

The HITL initialism traditionally refers to a test method whereby real hardware components are integrated into a simulation or emulation environment to check their behaviour in a controlled setting.

More recently, HITL has been adopted to mean human-in-the-loop, where humans actively participate in the training, evaluation, or operation of AI systems. HITL systems can learn and adapt to new situations more effectively by incorporating human feedback.

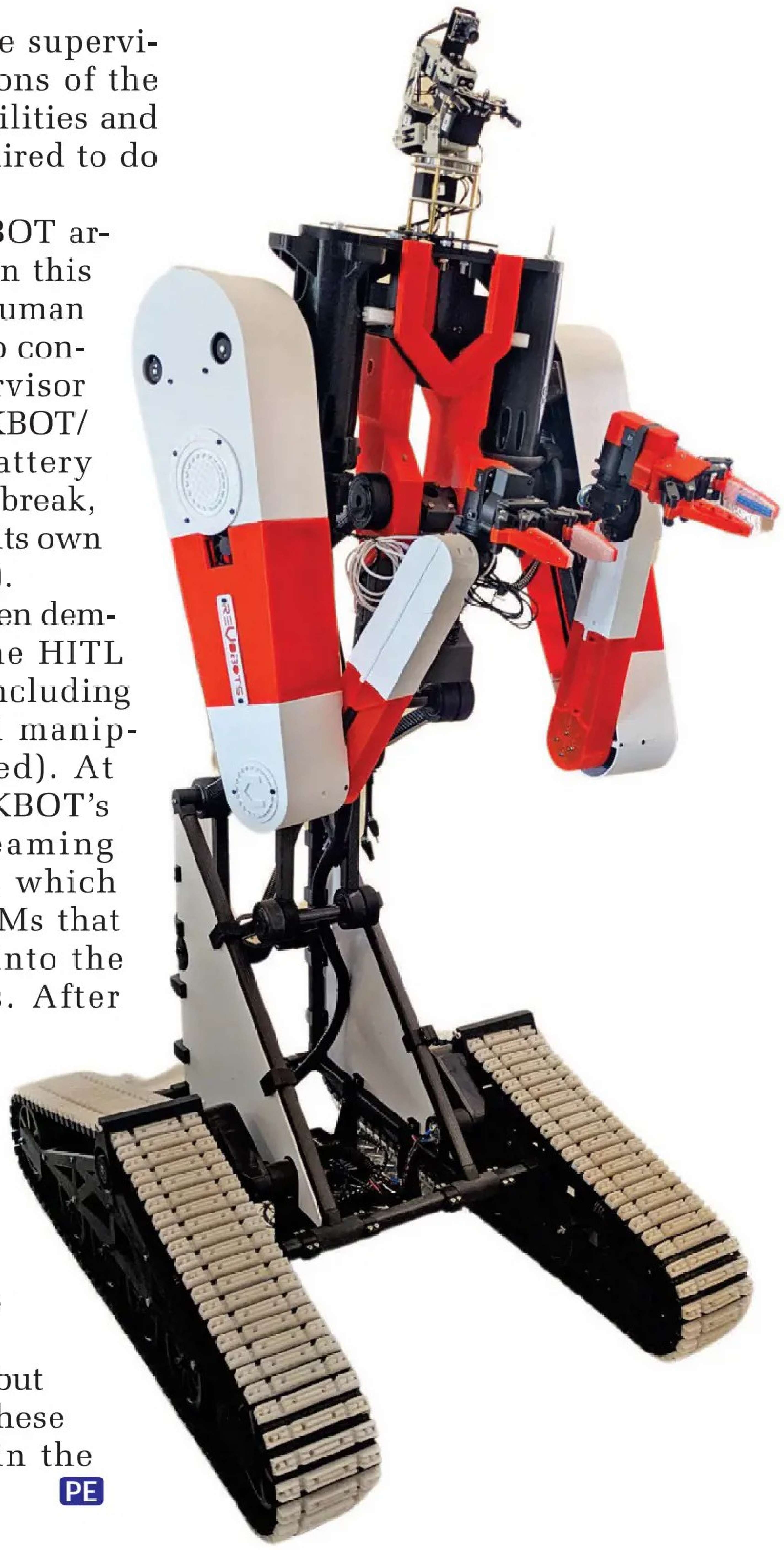
Picture the scene—a new human worker turns up at a factory to work on a production line. The supervisor will show the locations of the restroom and canteen facilities and demonstrate what is required to do the job.

Now, imagine a TASKBOT arriving in the same way. In this case, a remotely located human would use telepresence to control the robot. The supervisor would first guide the TASKBOT/human combo to the battery bank (like taking a coffee break, the TASKBOT can change its own batteries when necessary).

The supervisor would then demonstrate the job with the HITL replicating the actions (including recognising objects and manipulating them as required). At the same time, the TASKBOT's sensors would be streaming everything to the cloud, which would then train the SLMs that would be downloaded into the TASKBOT's processors. After that, the HITL would close the connection and then leave the TASKBOT to perform its duties.

I tell you, every day the world gets weirder and more wonderful at the same time.

I don't know about you, but I'm excited about what these robots could do for us in the near future! PE

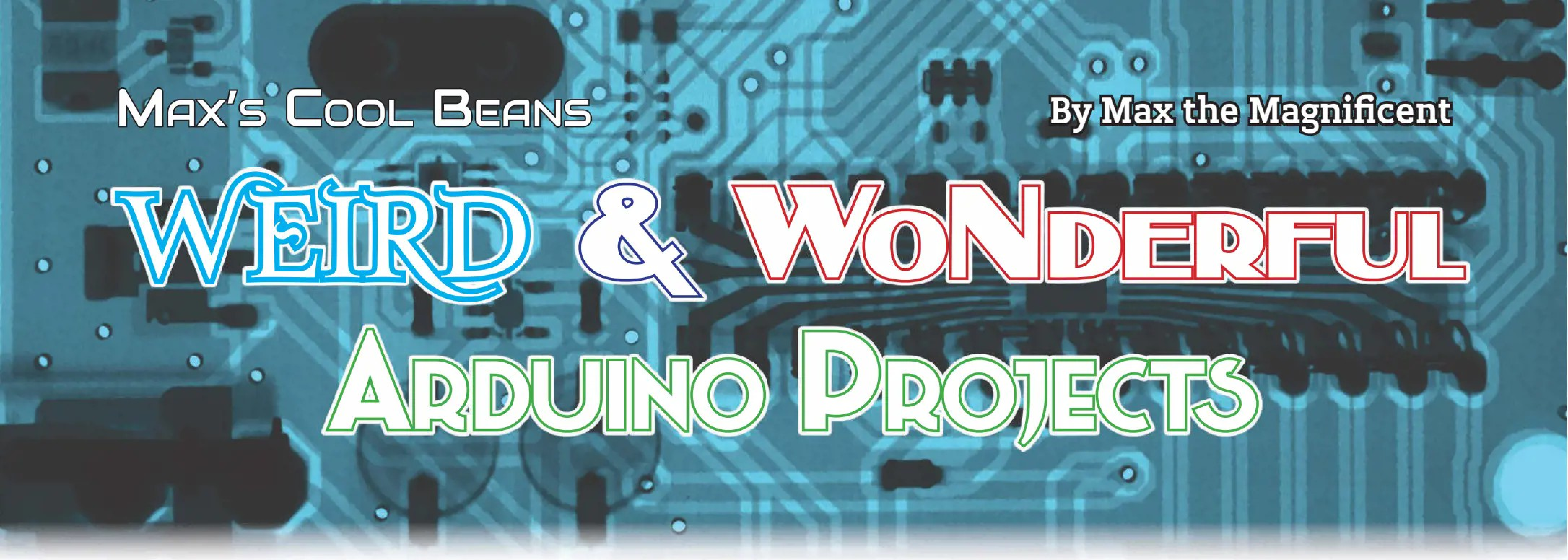


FIND ALL YOUR ELECTRONIC COMPONENTS IN ONE PLACE

BASIC MICRO UK
ELECTRONICS COMPONENT SUPPLIER
WWW.BASICMICRO.CO.UK

High-quality, genuine parts

01623 720730
Available from 9AM - 5PM Mon-Fri



Max's Cool Beans

By Max the Magnificent

Weird & Wonderful Arduino Projects

Part 7: predictably random

I'm so eager to plunge into the heart of this month's column that I can barely restrain myself. Let's take a moment to remind ourselves where we left things last time.

We started by adding a 74HC595 serial-in, parallel-out (SIPO) shift register integrated circuit (IC) to our breadboard. We connected this shift register's **SER** ('serial data') input to logic 1, and we used its outputs to drive the light-emitting diodes (LEDs) on our 8-segment bar graph display.

We also added three momentary pushbutton switches to the breadboard, along with corresponding pull-up and pull-down resistors. Initially, we used these switches to directly drive the shift register's **$\overline{\text{SRCLR}}$** ('shift register clear'), **SRCLK** ('shift register clock'), and **RCLK** ('output register clock') control pins. However, we soon ran into the problem of switch bounce.

To address the switch bounce problem, we disconnected the signals from the switches to the shift register and instead fed them to three of our Arduino Uno inputs. We then connected three of the Arduino's outputs to the inputs on the shift register (see Fig.1).

Next, we created a simple software routine to debounce the signals coming from the switches, and we used these debounced versions to control our shift register.

We had been using the three control signals individually. This meant that after pressing either of the **SRCLK** or **$\overline{\text{SRCLR}}$** switches to modify the contents of the shift register, we were obliged to press the **RCLK** switch to copy the new contents from the shift register into its output register.

The result was an excessive amount of switch pressing on our part. Thus, we concluded that column by noting that one of the advantages of running our signals through the Arduino is that we can write our software such that a single stimulus signal can be used to trigger multiple responses.

To cut a long story short (not how I usually like to do things), Fig.3 shows how we modified our switch debouncing code so that pressing the **SRCLK** switch resulted in a positive pulse (⎍) on the **SRCLK** signal to the shift register, followed by a positive pulse on the **RCLK** signal to the shift register.

Similarly, pressing the **$\overline{\text{SRCLR}}$** switch resulted in a negative pulse (⊔) on the **$\overline{\text{SRCLR}}$** signal to the shift register, followed by a positive pulse on the **RCLK** signal to the shift register.

You can further refresh your memory by downloading an image showing the current state of the breadboard in the file named **CB-july25-brd-01.pdf**, along with a copy of our latest and greatest program in the file named **CB-july25-code-01.txt**. These are found within the July 2025 download package on the *PE* website at **https://pemag.au/link/ac5q**.

Fabulous feedback

Feedback is a crucial concept in both analog and digital circuits. In the case of analog circuits, which work with continuous signals (that can take on a practically infinite number of values within a given range, and that change smoothly over time without steps), feedback refers to the process whereby a portion of the output signal is fed back into the input to affect the behaviour of the circuit.

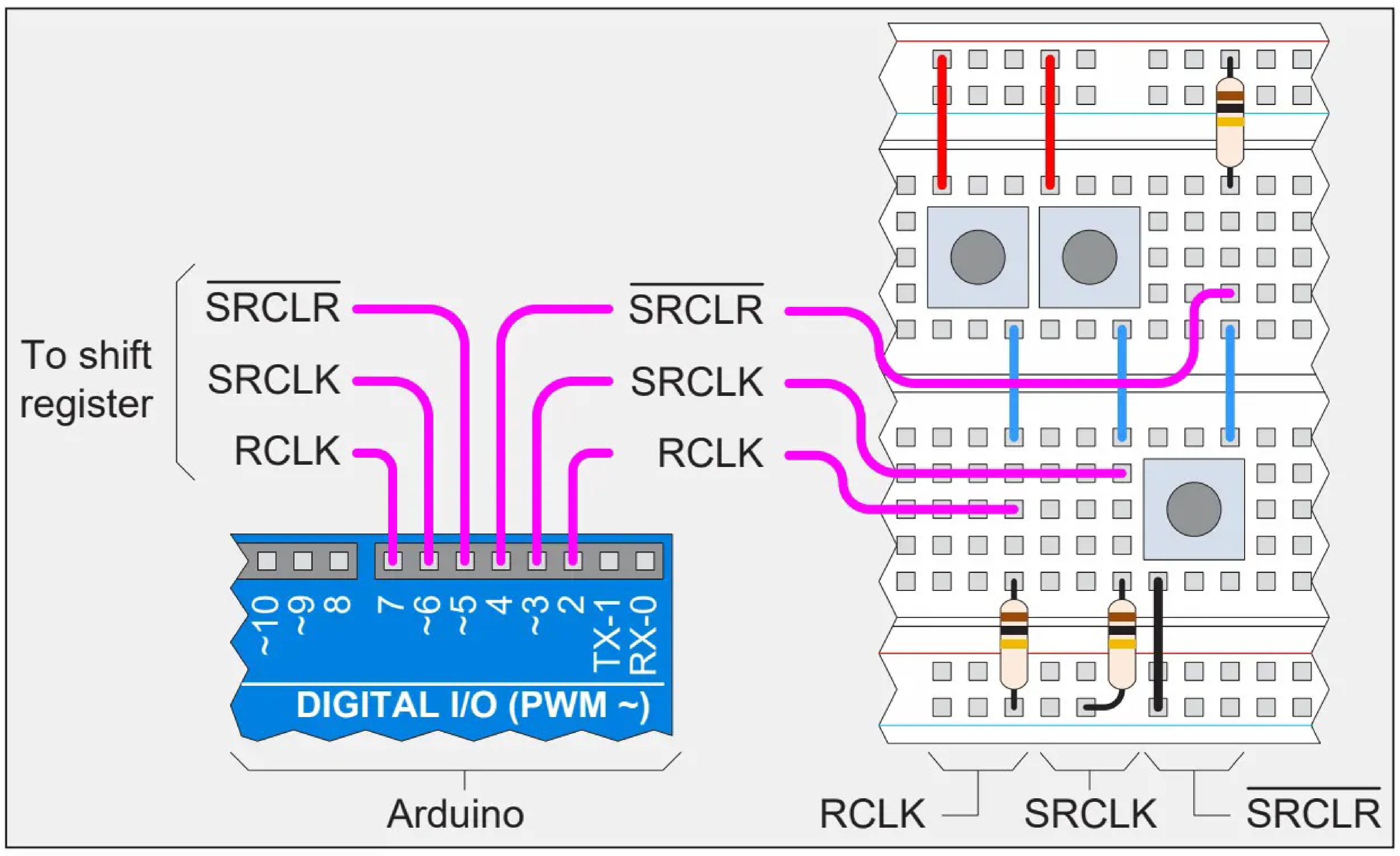


Fig.1. the current state of our breadboard.

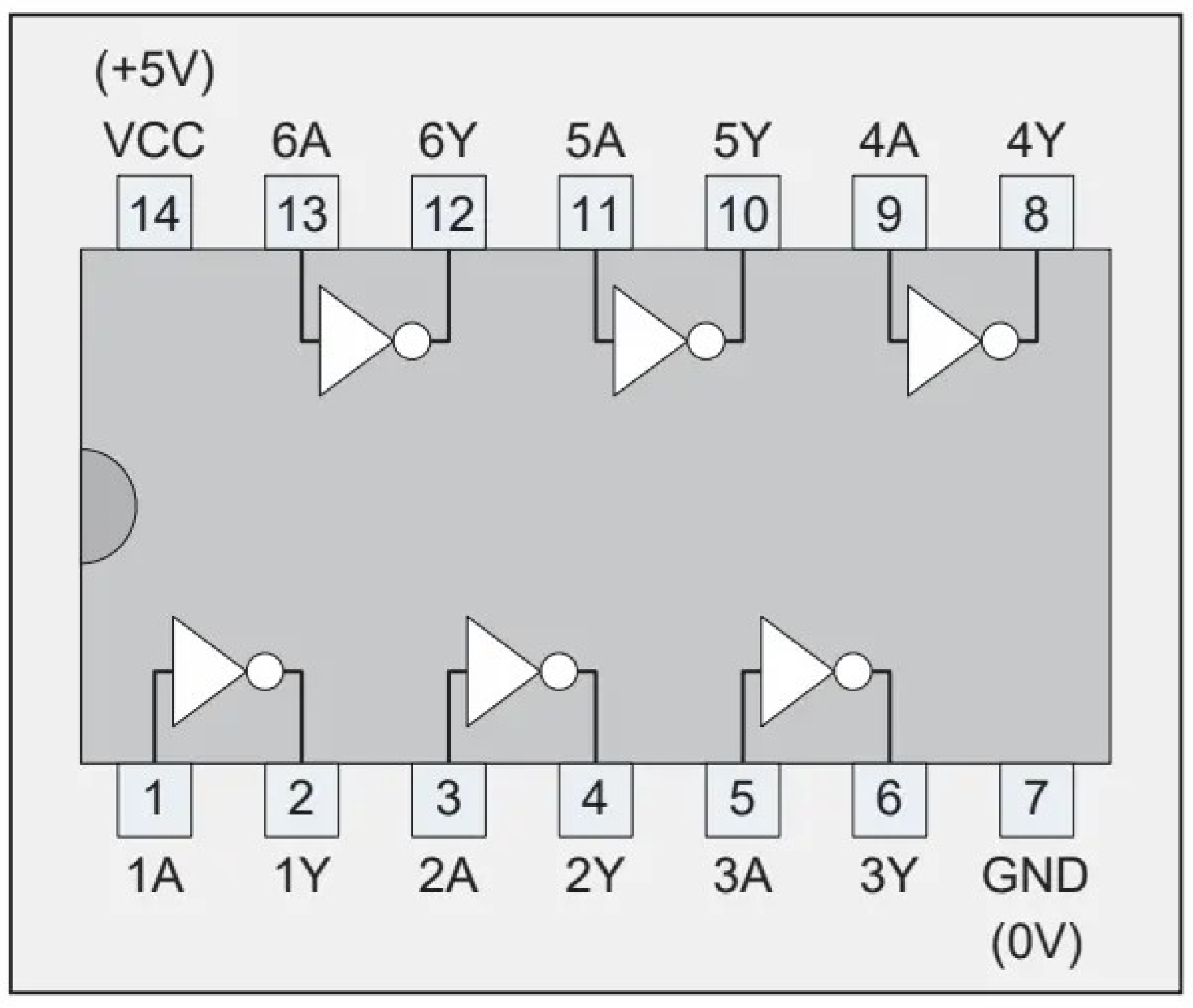


Fig.2: the 74xx04 IC contains six inverters.

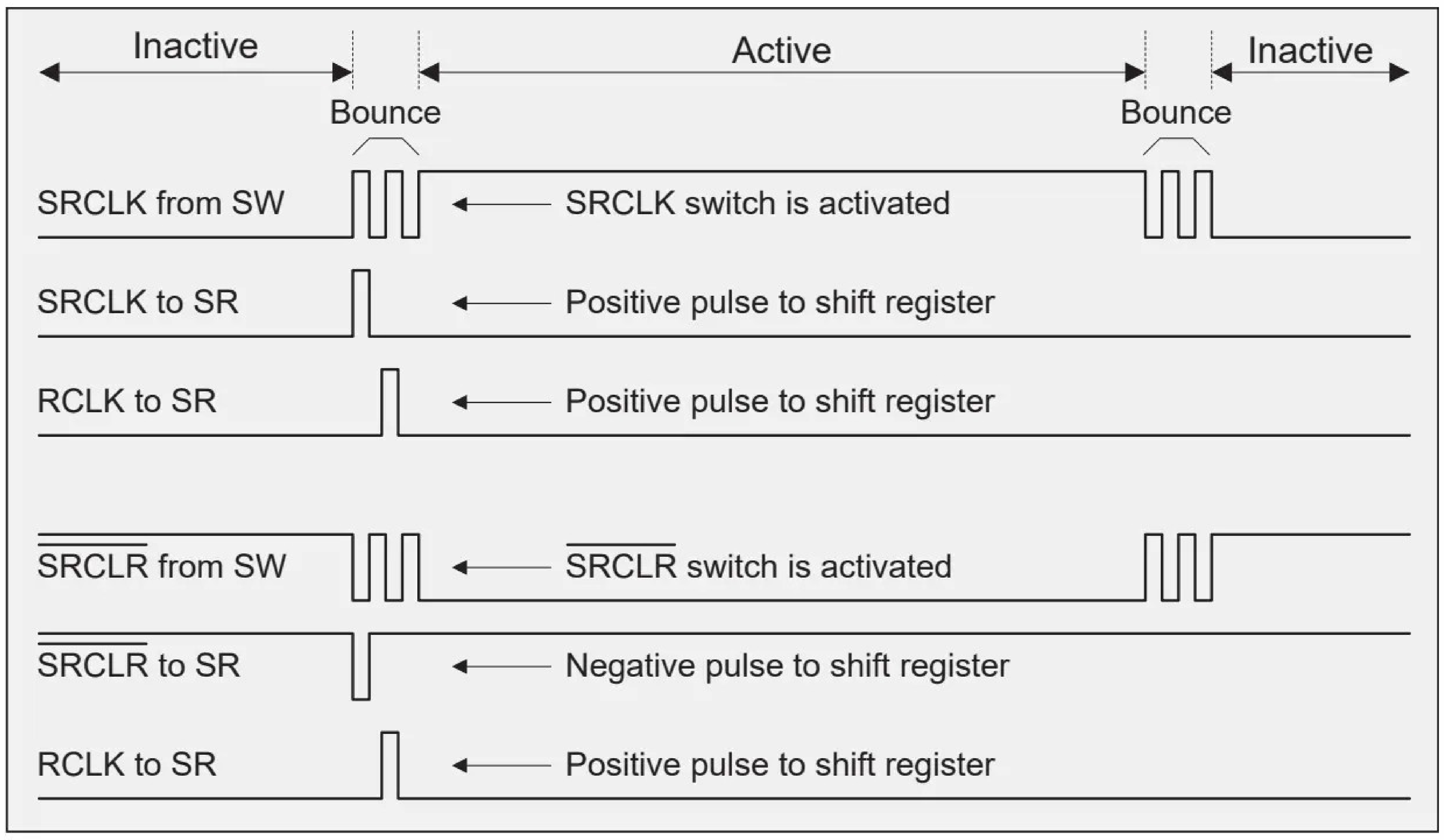

Inactive
Active
Inactive
Bounce
Bounce
SRCLK from SW
SRCLK switch is activated
SRCLK to SR
Positive pulse to shift register
RCLK to SR
Positive pulse to shift register
$\overline{\text{SRCLR}}$ from SW
$\overline{\text{SRCLR}}$ switch is activated
$\overline{\text{SRCLR}}$ to SR
Negative pulse to shift register
RCLK to SR
Positive pulse to shift register

Fig.3: with software, one stimulus can trigger multiple responses.

The most common type of feedback used in analog circuits is **negative feedback**, which is employed for tasks like stabilising the circuit, reducing distortion, improving bandwidth, and increasing linearity. By comparison, **positive feedback** amplifies the signal and can lead to instability, but it can also be used intentionally in circuits like oscillators, where it's needed to generate continuous waveforms.

In digital circuits, which typically work with discrete 0 (low) and 1 (high) signals, feedback is primarily used to implement and control the state of memory elements, like flip-flops, registers, counters, and more.

Consider Fig.4(a), with a constant '1' being fed into a shift register, which is the way the chip on our breadboard is currently configured. Now compare this with Fig.4(b), in which the output from the shift register is first inverted and then fed back into the input.

As an aside, whenever I see a shift register with feedback in this form, I cannot help but think of the Ouroboros, an ancient symbol depicting a snake (or a dragon) eating its own tail. Adopted as a symbol in both Gnosticism and Hermeticism, the Ouroboros is often interpreted as a symbol for eternal cyclic renewal or a cycle of life, death, and rebirth.

Bob would be proud

Way back in the mists of time that we used to call October 2024, in our *Arduino Bootcamp* column, we introduced the American inventor, engineer, computer pioneer, and professor, Robert Royce "Bob" Johnson (1928–2016). Amongst many other inventions and innovations, Bob came up with something we now call the Johnson counter.

In fact, Fig.4(b) is a classic implementation of a Johnson counter. While it's certainly fun to think about this sort of thing theoretically, it's even more enjoyable to see it in action, so that's what we will do.

Since I know more than my fair share of people called Bob, I cannot help but think of a Johnson counter as being a Bob counter. As part of our bootcamp columns, we introduced the 74xx04 IC, which contains six inverters, aka NOT gates (Fig.2).

The reason I'm using "xx" in the 74xx04 device name is that there are various 'flavours' of the chip that essentially do the same job. You could use a 74LS04 (low-power schottky TTL) or a 74HC04 (high-speed CMOS) version of the chip, or perhaps something else. Given a choice, the HC technology is better here because it's more tolerant of input voltage variations.

We are going to add one of these devices to our breadboard, as illustrated in Fig.5 (a corresponding image of the full board is available in the file named **CB-july25-brd-02.pdf**). Although I've shown this deployment in one fell swoop, it's best to take things one step at a time.

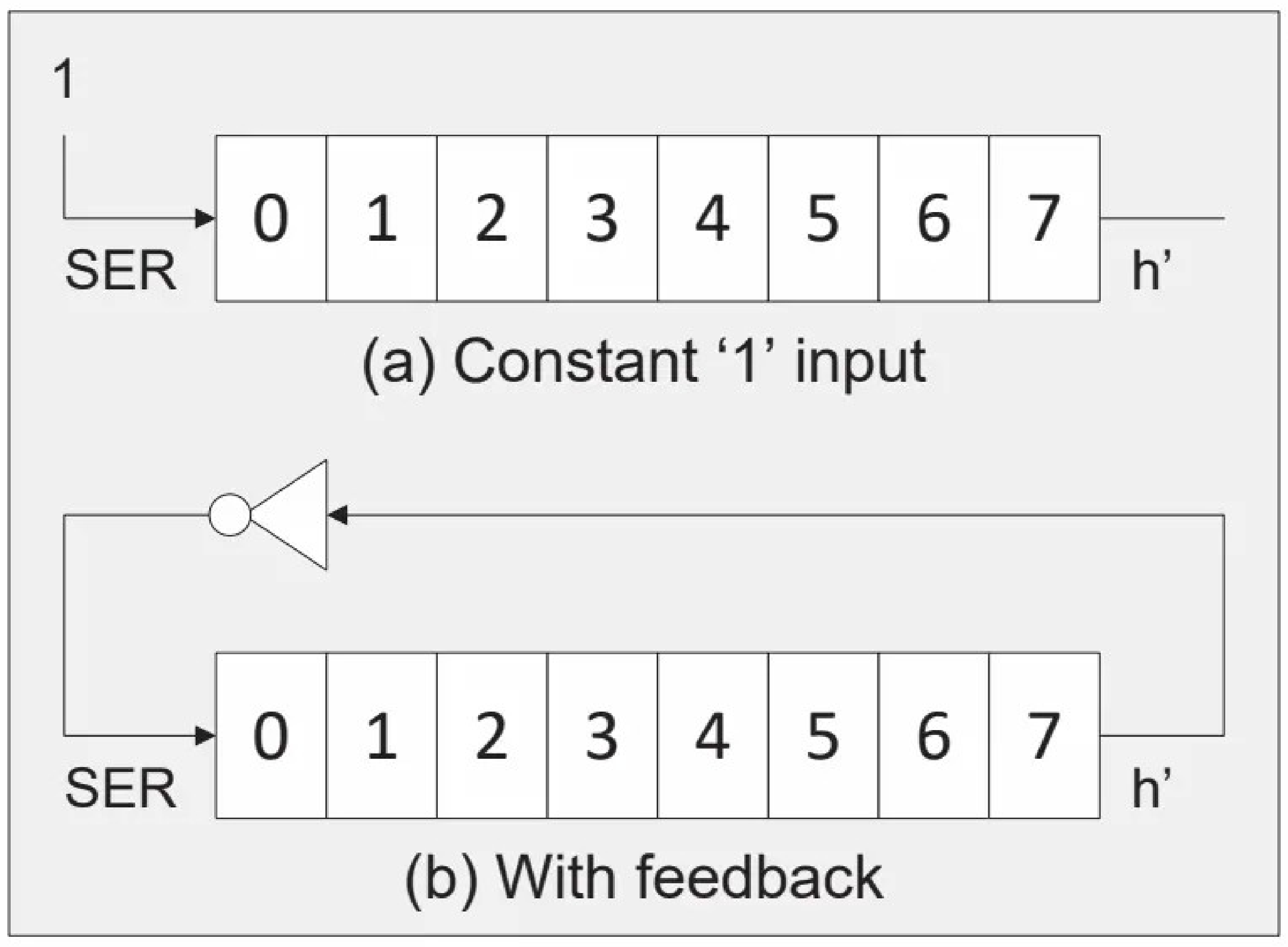
First, make sure everything is powered down, then add the 74xx04 to the breadboard along with its red (+5V) and black (0V) connections and associated ceramic capacitor. Power things up again and use your multimeter to verify that we see +5V (give or take) on pin 14 and 0V on pin 7.

The fastest and easiest way to check this is to set your multimeter to the 20V DC range (or whatever range is closest to, but higher than, 5V), then place the red and black probes on pins 14 and 7 of the 74xx04, respectively.

You should see a reading of +5V (give or take), indicating that all is well. If you see a reading of -5V (give or take), then either you have your probes on the wrong pins, your probes are plugged into the wrong input jacks on your multimeter, or you've connected the wires incorrectly on your breadboard.

If you see a reading of 0V, then one or both of your red and black wires are either broken, connected to the wrong IC pin, or connected to the wrong rail. A quick visual check may resolve the concern. If not, then proceed to the next step.

Hopefully, you already have header pins plugged into the power and ground rails somewhere on your breadboard. If not, add a couple as shown in Fig.5. **Do not** place these header pins next to (or even close to) each other, because


1
SER
0
1
2
3
4
5
6
7
h'
(a) Constant '1' input
SER
0
1
2
3
4
5
6
7
h'
(b) With feedback

◀ Fig.4: adding feedback to a shift register.

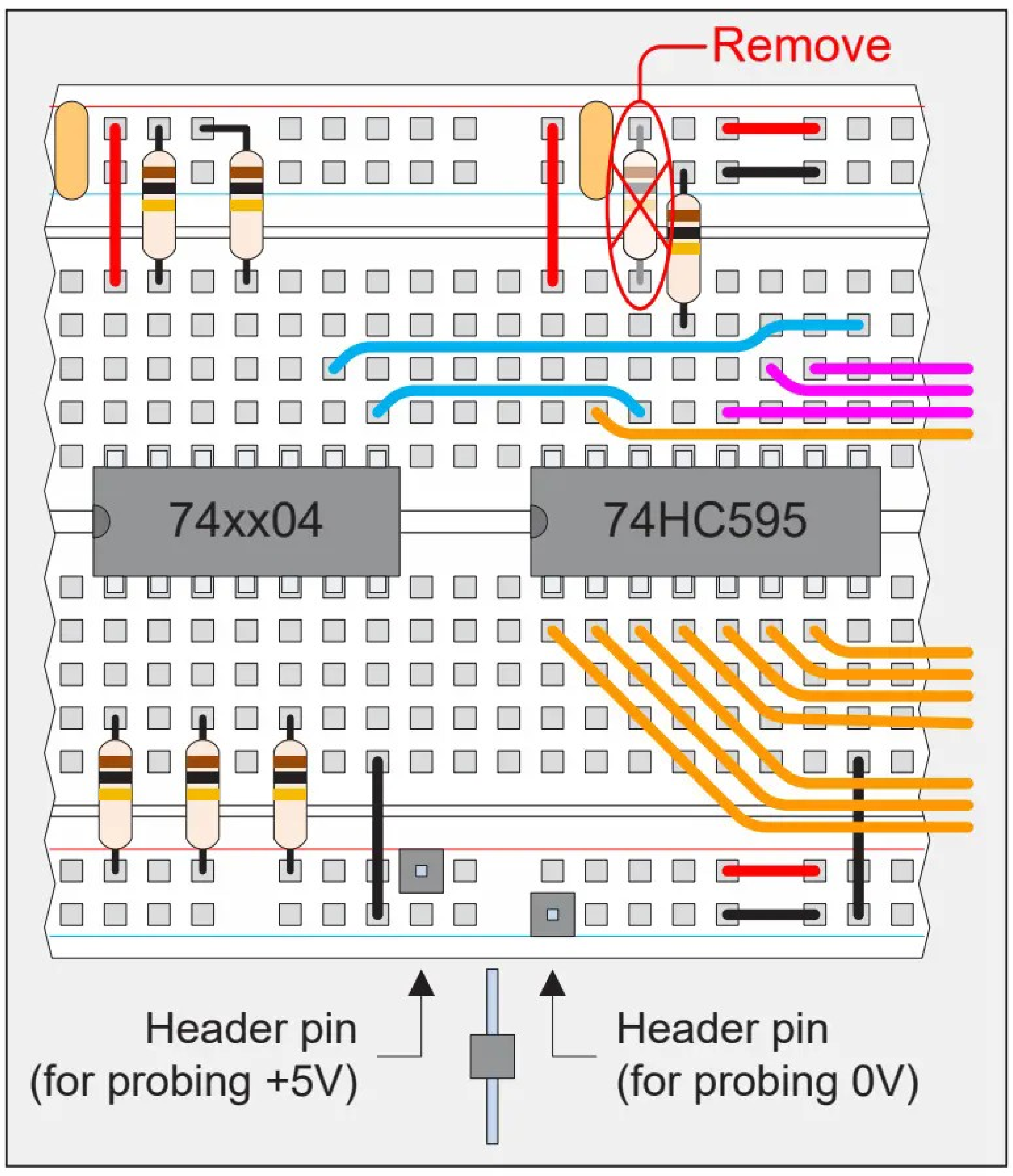
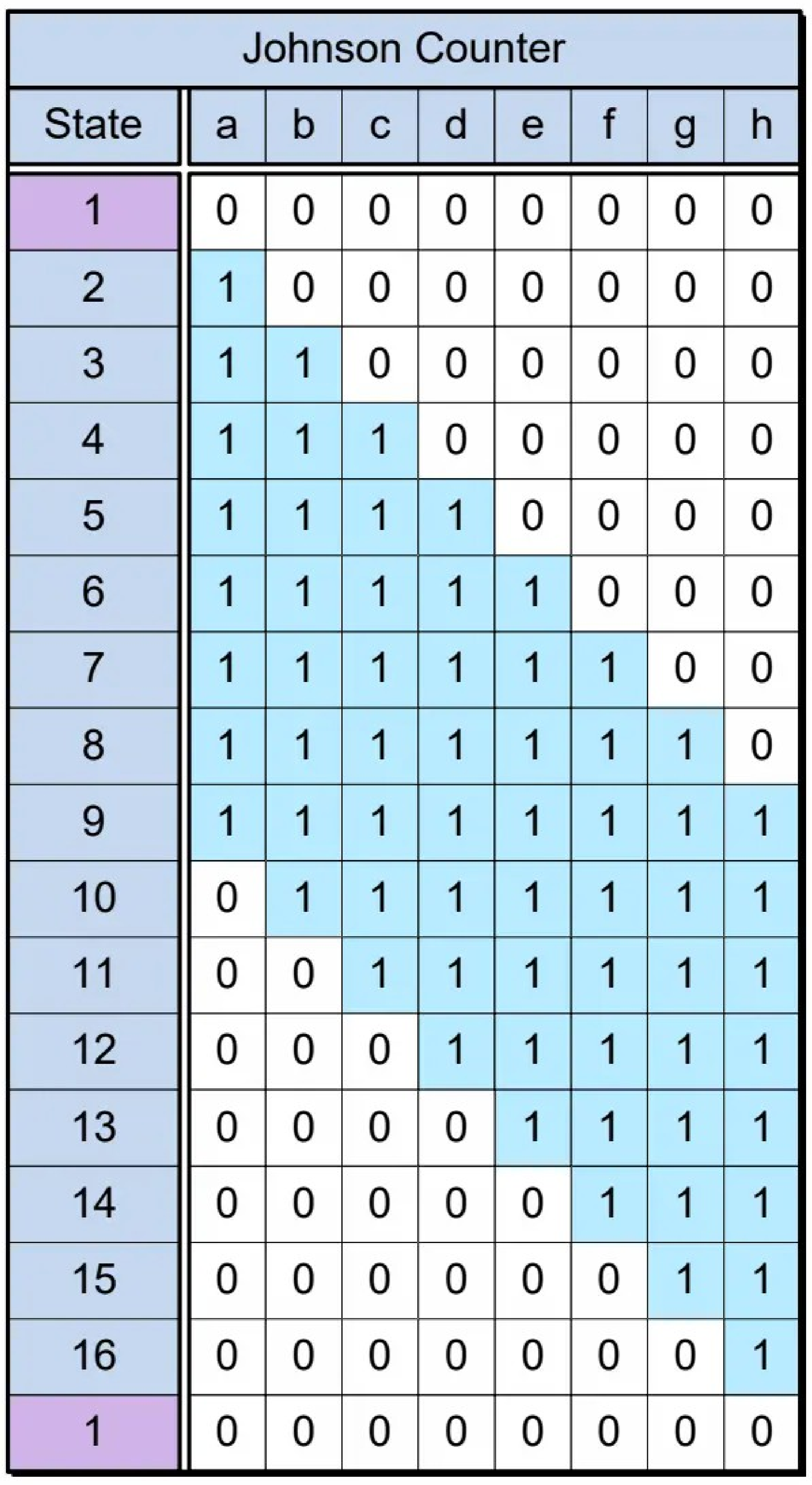

Remove
74xx04
74HC595
Header pin
(for probing +5V)
Header pin
(for probing 0V)

Fig.5: adding the 74xx04 to our breadboard. ▶



Johnson Counter								
State	a	b	c	d	e	f	g	h
1	0	0	0	0	0	0	0	0
2	1	0	0	0	0	0	0	0
3	1	1	0	0	0	0	0	0
4	1	1	1	0	0	0	0	0
5	1	1	1	1	0	0	0	0
6	1	1	1	1	1	0	0	0
7	1	1	1	1	1	1	0	0
8	1	1	1	1	1	1	1	0
9	1	1	1	1	1	1	1	1
10	0	1	1	1	1	1	1	1
11	0	0	1	1	1	1	1	1
12	0	0	0	1	1	1	1	1
13	0	0	0	0	1	1	1	1
14	0	0	0	0	0	1	1	1
15	0	0	0	0	0	0	1	1
16	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0

Fig.6: an 8-bit (16-state) Johnson counter.

it's easy to short them out with your multimeter probe if you aren't careful or if your attention wanders (don't ask me how I know this).

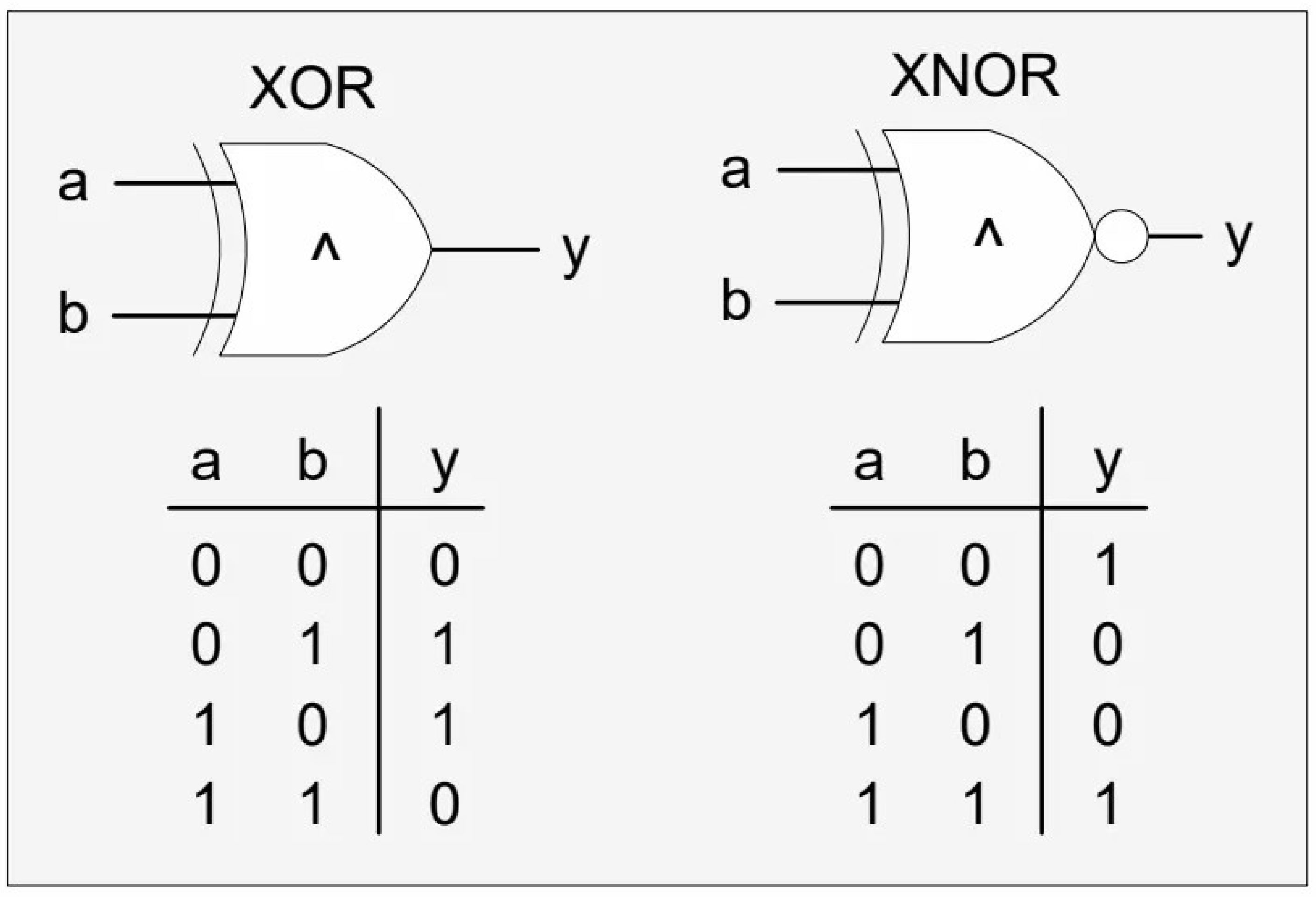
Connect the black probe to the 0V header pin and the red probe to pin 14 on the IC. If you don't see +5V (give or take), then power everything down, replace the red wire on your breadboard, and return to the beginning of the test procedure.

If you do see +5V, then place the red probe on the +5V header pin and the black probe on pin 7 on the IC. If you don't see +5V (give or take), then power everything down, replace the black wire on your breadboard, and return to the beginning of the test procedure. Once you are happy that the chip is powered as expected, power things down once more.

We are going to use the NOT gate with its input and output at pins 9 and 8 on the 74xx04, respectively. It's always good practice to make sure that any unused inputs to logic devices have associated pull-up or pull-down resistors (see my *How to Handle the Inputs to Unused Logic Gates* column for more details; pemag.au/link/ac55).

So, let's add five 10kΩ pull-up resistors to the inputs of the unused gates on pins 1, 3, 5, 11, and 13, ensuring that these gates cannot start oscillating or draw excessive power.

You may call me overcautious (I've been called worse, and by my dear old mum at that), but it really wouldn't hurt to power things up, connect your black multimeter probe to IC pin 7 (or to the 0V header pin), use your red probe to verify that you see +5V (give or take) on pins 1, 3, 5, 11 and 13, then power everything down again.



a	b	y
0	0	0
0	1	1
1	0	1
1	1	0

a	b	y
0	0	1
0	1	0
1	0	0
1	1	1

Fig.7: symbols and truth tables for XOR and XNOR gates.

We're almost there. Connect the shift register's **h'** output (pin 9 on the 74HC595) to the NOT gate input (pin 9 on the 7404). Pay attention! This can be a tad confusing because the 74HC595 and 74xx04 chips are in 16-pin and 14-pin packages, respectively, which means their pin 9s are in different locations.

Finally, remove the resistor that's currently connecting the shift register's **SER** ('serial data') input (pin 14 on the 74HC595) to logic 1, then connect the NOT gate output (pin 8 on the 7404) to the shift register's **SER** input (pin 14 on the 74HC595).

Are you ready to rock and roll? I know I am!

Bob's your uncle

Power everything up. As we discussed last month, according to its data sheet (pemag.au/link/ac10), the 74HC595 doesn't contain any special power-on-reset (PoR) functionality to clear its internal registers and initialise them with values of 0.

Thus, every time we apply power to the breadboard, we may end up with the chip's output register containing a random collection of 0s and 1s, resulting in corresponding off and on LED segments that are different every time power is removed and reapplied.

Similarly, the chip's internal shift register, whose contents we won't see until we copy them to the output register, may also contain a random assortment of 0s and 1s. So, just to be on the safe side, press and release the $\overline{\textbf{SRCLR}}$ (right-hand) switch.

As we discussed earlier (Fig.3), our current program will use this event to present a negative pulse (⊔) on the $\overline{\textbf{SRCLR}}$ signal to the shift register, followed by a positive pulse on the **RCLK** signal to the shift register, thereby ensuring that all register bits contain 0.

Now repeatedly press and release the **SRCLK** switch and watch the 8-bit (16-state) Johnson counter sequence unfold on the bar graph display, as illustrated in Fig.6 (the 0s and 1s in the table correspond to LED segments on the display being off and on, respectively).

Personally, I think this is awesome. I've just spent more time than might seem reasonable playing with this little beauty. But I can hear you crying, "I want more!" Well, never let it be said that I failed to serve.

Meet the LFSR

A linear feedback shift register (LFSR) is a shift register that operates with a linear function of its previous state to generate a sequence of bits. These functions are commonly used for tasks like pseudo-random number generation, cryptography, error detection and data scrambling.

At the heart of an LFSR is a series of flip-flops that shift their contents at each clock cycle, just like our 74HC595. However, unlike our Johnson counter, in which we fed the inverted version of the last bit back into the input, in the LFSR, multiple register bits (taps) are XORed (or XNORed) together and fed back to the input.

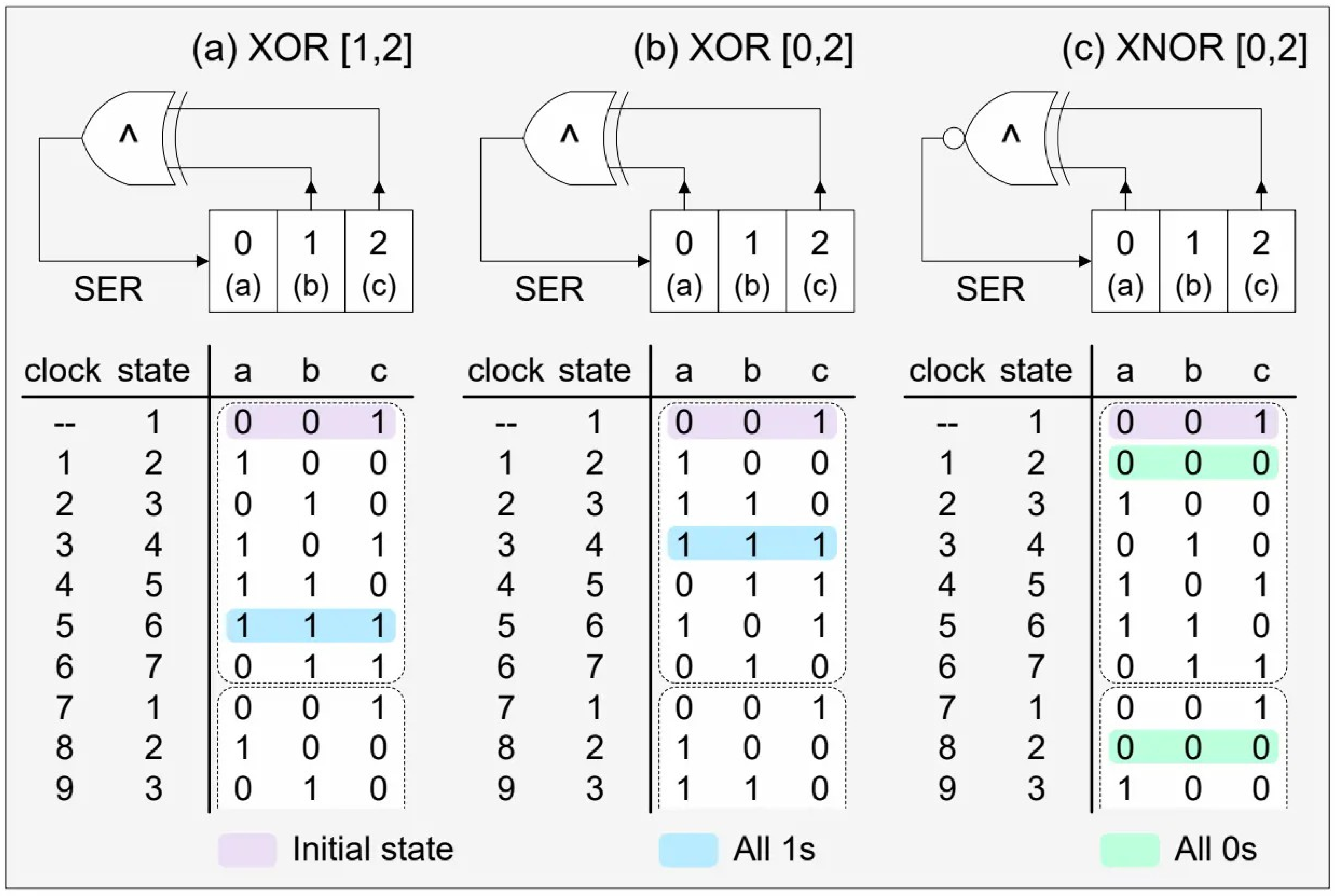
Before we proceed, let's briefly remind ourselves as to the symbols and truth tables associated with an XOR ('exclusive OR') gate and its inverting counterpart, the XNOR ('exclusive NOR). These are illustrated in Fig.7 (I'll explain why it's "XNOR" rather than "NXOR" in my next column when we delve deeper into logic functions).

Observe the 'bobble' (or 'bubble') on the output of the XNOR symbol. This indicates an inverted output, which, in this case, means the output is inverted compared to that from the XOR.

To help us wrap our brains around this, let's start by assuming that we are working with a small collection of 3-bit shift registers (Fig.8). As we know, all the register bits share a common clock, which we've omitted from these diagrams to keep things simple.

The register bits are numbered 0, 1, and 2. Alternatively, we can think of these as being named **a**, **b**, and **c** to match the outputs from our 74HC595 chip. For the purposes of this portion of our discussions, let's assume the register is loaded with an initial value of 001 in binary.

In our first example, shown in Fig.8(a), the tap points are bits 1 & 2, which we can refer to as [1,2]. Also, we



(a) XOR [1,2]

clock	state	a	b	c
--	1	0	0	1
1	2	1	0	0
2	3	0	1	0
3	4	1	0	1
4	5	1	1	0
5	6	1	1	1
6	7	0	1	1
7	1	0	0	1
8	2	1	0	0
9	3	0	1	0

(b) XOR [0,2]

clock	state	a	b	c
--	1	0	0	1
1	2	1	0	0
2	3	1	1	0
3	4	1	1	1
4	5	0	1	1
5	6	1	0	1
6	7	0	1	0
7	1	0	0	1
8	2	1	0	0
9	3	1	1	0

(c) XNOR [0,2]

clock	state	a	b	c
--	1	0	0	1
1	2	0	0	0
2	3	1	0	0
3	4	0	1	0
4	5	1	0	1
5	6	1	1	0
6	7	0	1	1
7	1	0	0	1
8	2	0	0	0
9	3	1	0	0

Fig.8: alternative tap and feedback functions for LFSRs.

are using an XOR gate to implement our feedback function. Observe that the register cycles through all possible combinations of 0 and 1s, **including** 111, but **excluding** 000.

In the case of our second example, shown in Fig.8(b), we stick with our XOR feedback function from Fig.8(a), but we use taps of [0,2]. As we see, the Fig.8(b) implementation follows a different sequence to its Fig.8(a) counterpart, but it still cycles through every combination of 0s and 1s, excluding 000.

Finally, in the case of our third example shown in Fig.8(c), we keep our [0,2] taps from Fig.8(b), but exchange the XOR for an XNOR to implement our feedback function. In this case, the register follows a different sequence, cycling through all possible combinations of 0 and 1s, **excluding** 111, but **including** 000.

A binary field with **n** bits can assume 2^n unique combinations of 0s and 1s; for example, a 3-bit field can support $2^3 = 8$ different values. The sequence depends on the tap points we select. In many cases, an LFSR will end up cycling around a loop comprising a limited number of states.

In some cases, the LFSR will cycle through $(2^n - 1)$ states, which means every possible state apart from all 0s with an XOR feedback path or all 1s with an XNOR feedback path. In such a case, it's classed as a **maximal-length** LFSR. Since all three examples in Fig.8 repeatedly cycle through $(2^3 - 1) = 7$ states, they are indeed maximal-length LFSRs.

Implementing our own LFSR

What we are going to do now is use our 74HC595 8-bit shift register to implement our own LFSR. Unfortunately, we have a couple of teeny-tiny problems. Let's start with the fact that each LFSR supports some number of tap combinations that will result in maximal length sequences and many combinations that… don't. The trick is weeding out the one we want.

Many moons ago, I was happy to find a book that provided tables detailing maximal length LFSR tap points for registers with different numbers of bits. After some investigation, I was disappointed to discover that many of these tap combinations were… well, not to put too fine a point on it… wrong.

As a result, I ended up writing a C program to generate my own humble tabular offering (I pride myself on my humility), which I documented in my *Bebop to the Boolean Boogie* book.

The next problem we have is that the minimal number of taps for an 8-bit maximal length LFSR is four. One such tap combination is [1,2,3,7], but this would require us to use three XOR gates.

That wouldn't normally be a problem per se, but—for reasons to be disclosed in the future—I wish to use only a single XOR or XNOR gate.

Thus, what we are going to do is use our 74HC595 to create a 7-bit LFSR using two taps at [0,6], as per Fig.9.

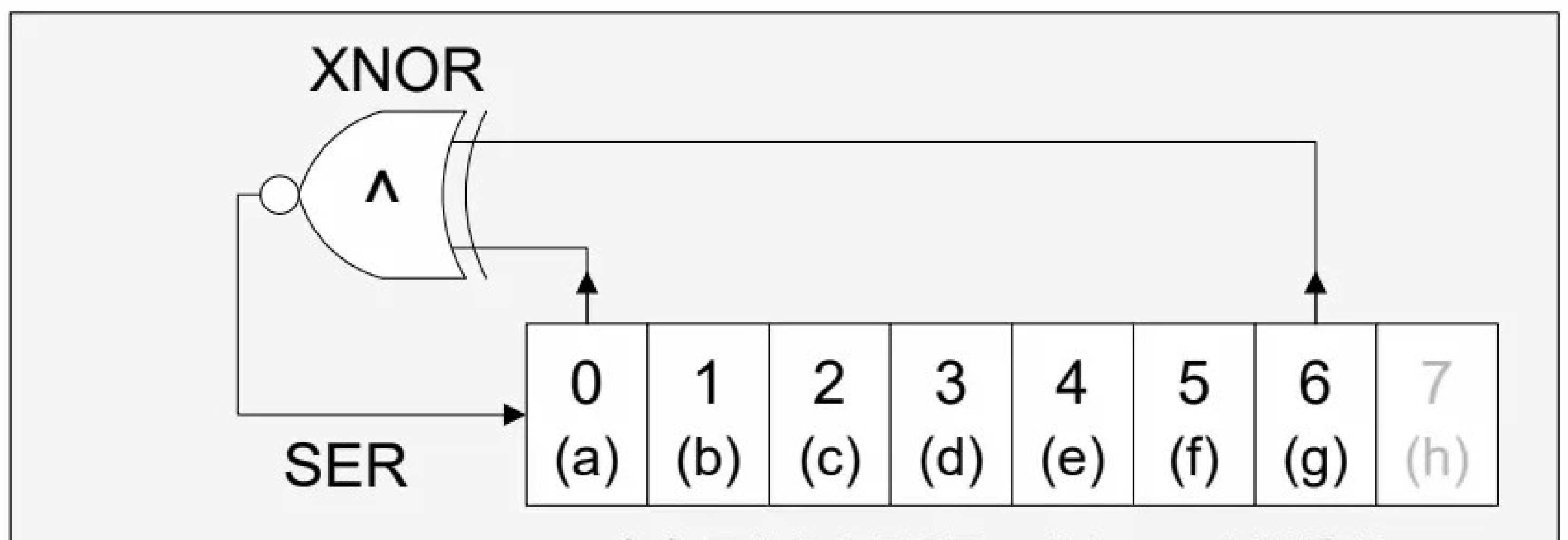
We are not interested in poor old bit 7 in for this implementation, so it will simply be a copy of whatever used to be in bit 6 from the previous clock. If you find this distracting, you have three main options:

1. Live with the distraction.
2. Remove the series resistor associated with bit 7 of the bar graph display.
3. Cover the offending segment on the display with a bit of tape (that's what I did).

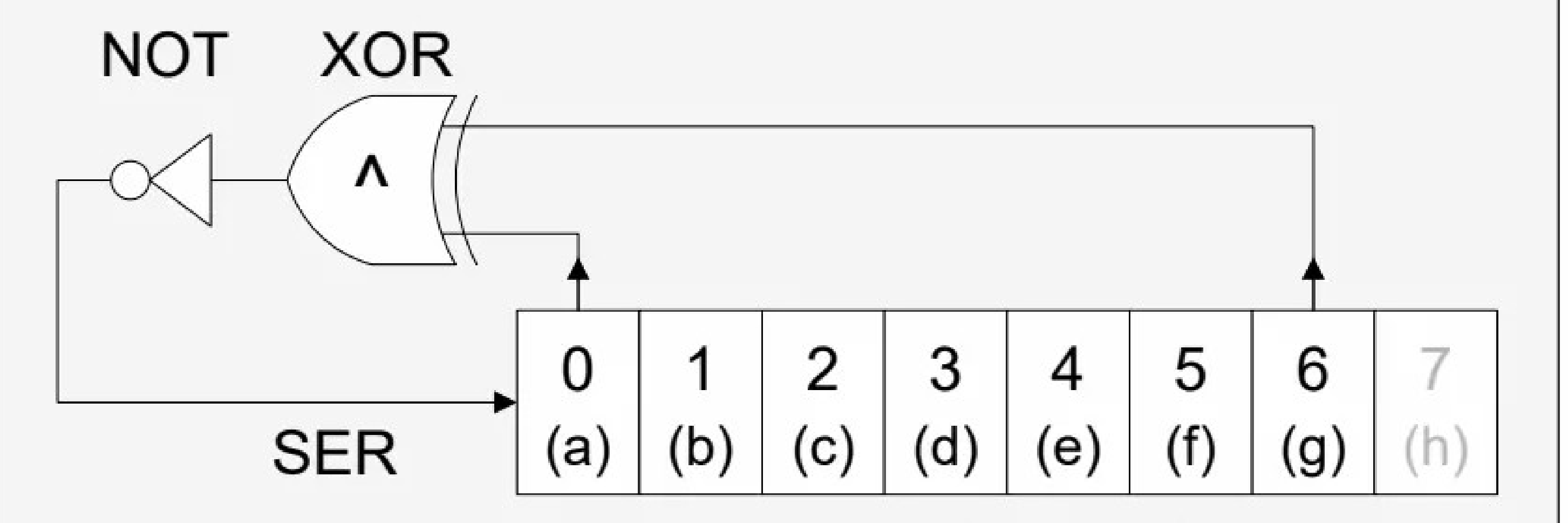
The reason we wish to use an XNOR gate rather than XOR is that, when we initialise our shift register, it is cleared to all 0s. If we used an XOR gate in our feedback path, that's the way it would stay, because 0 XOR 0 = 0. By comparison, 0 XNOR 0 = 1, which will get us out of the initial all-0s state.

This is where we come to the second problem, which is that we don't actually have an XNOR gate at our disposal. Such chips are available, but we don't have one in our arsenal of popular SN74xx00 chips that we previously purchased as part of the *Arduino Bootcamp* series (**pemag.au/link/ac2k**).

What we do have in that kit is a 74xx86, which is a quad 2-input XOR chip (Fig.10). We can implement the



(a) 7-bit LFSR with an XNOR.



(b) XOR + NOT = XNOR

◀ Fig.9: a 7-bit LFSR only requires one XOR gate.

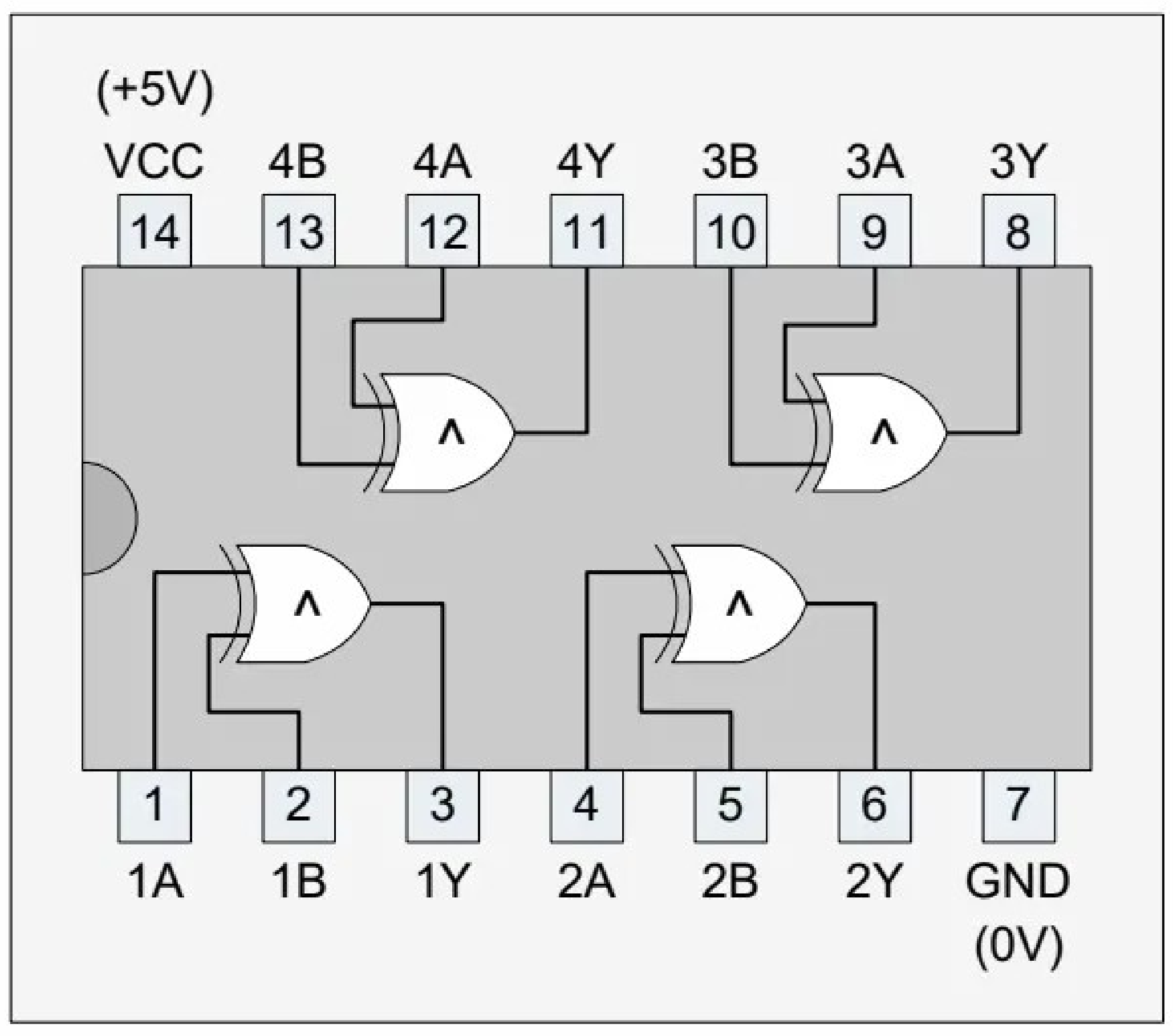


Fig.10: the ▶ 74xx86 quad XOR IC.

XNOR function by simply inverting the output from an XOR, as illustrated in Fig.9(b). Fortunately, we already have our handy-dandy 74xx04 hex inverter (NOT) chip ready and waiting for us on our breadboard.

If you didn't purchase the kit of SN74xx00 chips but instead opted to buy only the 74xx04 (hex inverter) and 74xx08 (quad 2-input AND) we used as part of the clock we constructed in our *Arduino Bootcamp* series, then… you're up the proverbial creek without a paddle.

I'm joking; there is a solution based on the parts we already have, but you'll need to wait until our next column to find out more. In the meantime, you'll just have to read along with the rest of us. Or go out and buy a 74xx86 chip (most large electronics retailers will sell at least one variant).

Assuming you have the 74xx86, add it to your breadboard as illustrated in Fig.11 (a depiction of the full board is available in the file named **CB-july25-brd-03.pdf**).

Do this in stages, like we did for the 74xx04. Start by adding the 74xx86 with its power and ground connections (and ceramic capacitor), and verify everything is as expected.

Next, add 10kΩ pull-up resistors to the unused gate inputs. We're going to use gate 2 in Fig.10, so add pull-ups to gate 1 (pins 1 and 2), gate 3 (pins 9 and 10, and gate 4 (pins 12 and 13). While it might work even if you don't do this, we want to adopt good habits (we already have enough bad ones).

Now we need to connect our tap points [0,6] from the 74HC595 to the inputs of our XOR. It doesn't matter which of the taps is connected to which of the XOR inputs. I connected pins 15 and 6 (corresponding to taps 0 and 6) on the 74HC595 to pins 4 and 5 on the 74xx86, respectively.

As illustrated in Fig. 11, remove the wire connecting pin 9 of the 74HC595 to pin 9 on the 74xx04. Again, this can be a tad confusing because the 74HC595 and 74xx04 chips are in 16-pin and 14-pin packages, which means their pin 9s are in different locations.

Pin 9 on the 74HC595 is its **h'** output, which we were using to implement the Johnson counter and which we can think of as tap [7], although we aren't going to use it as such. Meanwhile, pin 9 on the 74xx04 is the input to the NOT gate.

Now connect the output from the XOR (pin 6 on the 74xx86) to the input of the NOT gate (pin 9 on the 74xx04). Happily, the output from the NOT gate (pin 8 on the 74xx04) is already connected to the **SER** input on the shift register (pin 14 on the 74HC595) from our previous experiment.

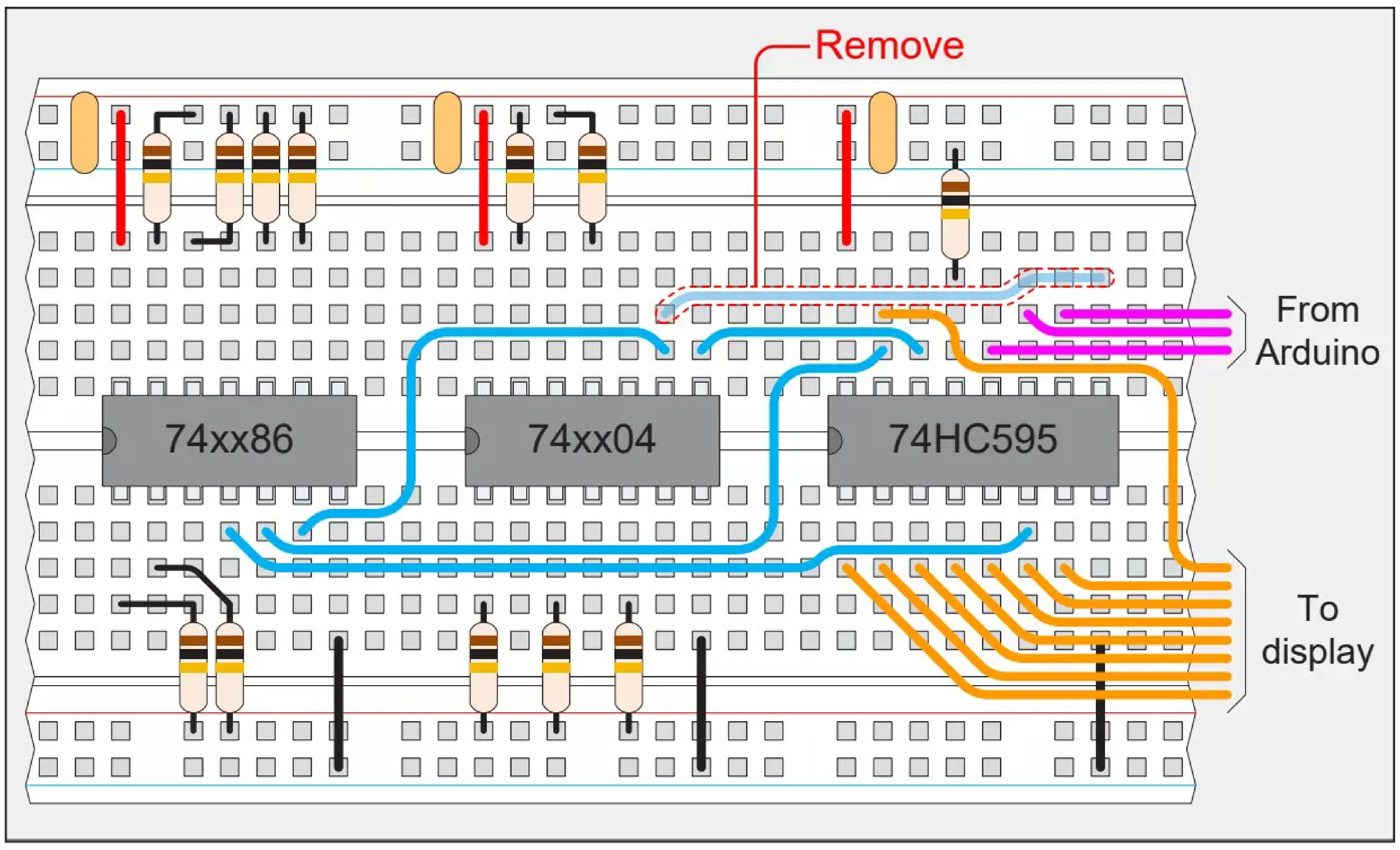


Fig.11: adding the 74xx86 quad XOR chip to our breadboard to make the LFSR.

Testing our LFSR

Power everything up and then press and release the **$\overline{\text{SRCLR}}$** (right-hand) switch, thereby clearing our LFSR to contain all 0s.

Now, press and release the **SRCLK** (middle) switch 16 times, pausing after each press to verify that the pattern of segments on the display matches the corresponding pattern of 0s and 1s shown in Fig.12.

I created this table in my office at work before verifying it on the hardware at home. Before you click further, why don't you use a pencil and paper to calculate the next six values (states 18 through 23), and then press and release the **SRCLK** button six more times, checking your work along the way?

Even though we are using only one 2-input XNOR gate in our feedback path, this still requires some mental gymnastics (so make sure you limber up first).

Observe the values in the "Decimal" column in Fig.12. These are the decimal equivalents of the binary 0s and 1s. They make it easy to see the pseudo-random nature of the numbers being generated.

'Pseudo-random' refers to something that appears random, but that is actually deterministic, meaning it follows a predictable sequence based on an initial 'seed' or starting point. In this case, our seed is the all 0s value.

Press and release the **$\overline{\text{SRCLR}}$** switch to reset our LFSR to its initial value. Then, repeatedly press and release the **SRCLK** button, and you will see that the same sequence of 'random-ish' numbers is generated.

We expect this to be a maximal-length implementation, but we should make sure that this is indeed the case. Since we are working with a 7-bit LFSR, we expect to see ($2^7 – 1$) = 127 unique states.

Use the **$\overline{\text{SRCLR}}$** switch to reset the LFSR, then repeatedly press and release the **SRCLK** button, counting each press as you go, and verify that the LFSR returns to its all 0s state on the count of 127.

Suppose we were to change our taps from [0,6] to [1,6]. Would this also result in a maximal-length implementation? I don't know. But I do know how to find out…

Homework assignment #1

Our current implementation utilises a 2-input XNOR function, as illustrated in Fig.9(a). We implemented this function using a combination of an XOR gate and a NOT gate, as in Fig.9(b). I have two tasks for you to ponder before we meet again: one easy and one… not.

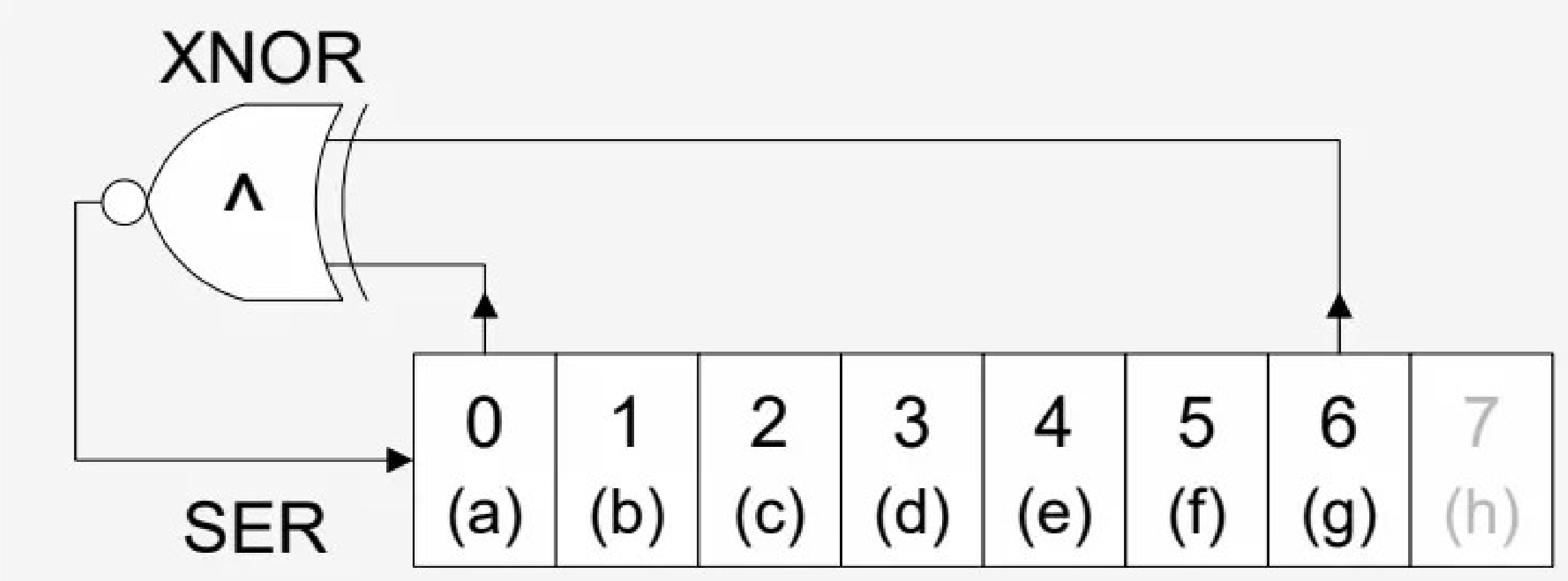
Easy: suppose we had only our 74xx86 chip containing its four 2-input XOR gates (so, no 74xx04 with its six NOT gates, and no 74xx08 with its four 2-input AND gates). Can you think of a way to implement the desired XNOR function?

Not easy: suppose we had only our 74xx04 with its six NOT gates and our 74xx08 with its four 2-input AND gates (so, no 74xx86 with its four 2-input XOR gates). Can you think of a way to implement the desired XNOR function?

Take a stab at these provocative posers and see how well you do. I will reveal all when next we meet.

What's that noise?

Now it's time to revisit the problem of the unwanted bouncing associated with our switches. When we introduced our initial switch debounce solution last month (as per the file named **CB-july25-code-01.txt**), I wrote, "In an



		2^6	2^5	2^4	2^3	2^2	2^1	2^0	
		64	32	16	8	4	2	1	
clock	state	a	b	c	d	e	f	g	decimal
--	1	🟩 0	0	0	0	0	0	0	0
1	2	1	0	0	0	0	0	0	64
2	3	0	1	0	0	0	0	0	32
3	4	1	0	1	0	0	0	0	80
4	5	0	1	0	1	0	0	0	40
5	6	1	0	1	0	1	0	0	84
6	7	0	1	0	1	0	1	0	42
7	8	1	0	1	0	1	0	1	85
8	9	1	1	0	1	0	1	0	106
9	10	0	1	1	0	1	0	1	53
10	11	0	0	1	1	0	1	0	26
11	12	1	0	0	1	1	0	1	77
12	13	1	1	0	0	1	1	0	102
13	14	0	1	1	0	0	1	1	51
14	15	0	0	1	1	0	0	1	25
15	16	0	0	0	1	1	0	0	12
16	17	1	0	0	0	1	1	0	70
:	:	:	:	:	:	:	:	:	:
125	126	0	0	0	0	0	1	1	3
126	127	0	0	0	0	0	0	1	1
127	1	🟩 0	0	0	0	0	0	0	0
128	2	1	0	0	0	0	0	0	64
:	:	:	:	:	:	:	:	:	:

🟩 Initial state and all 0s

Fig.12: the sequence for a 7-bit XNOR LFSR with [0,6] taps.

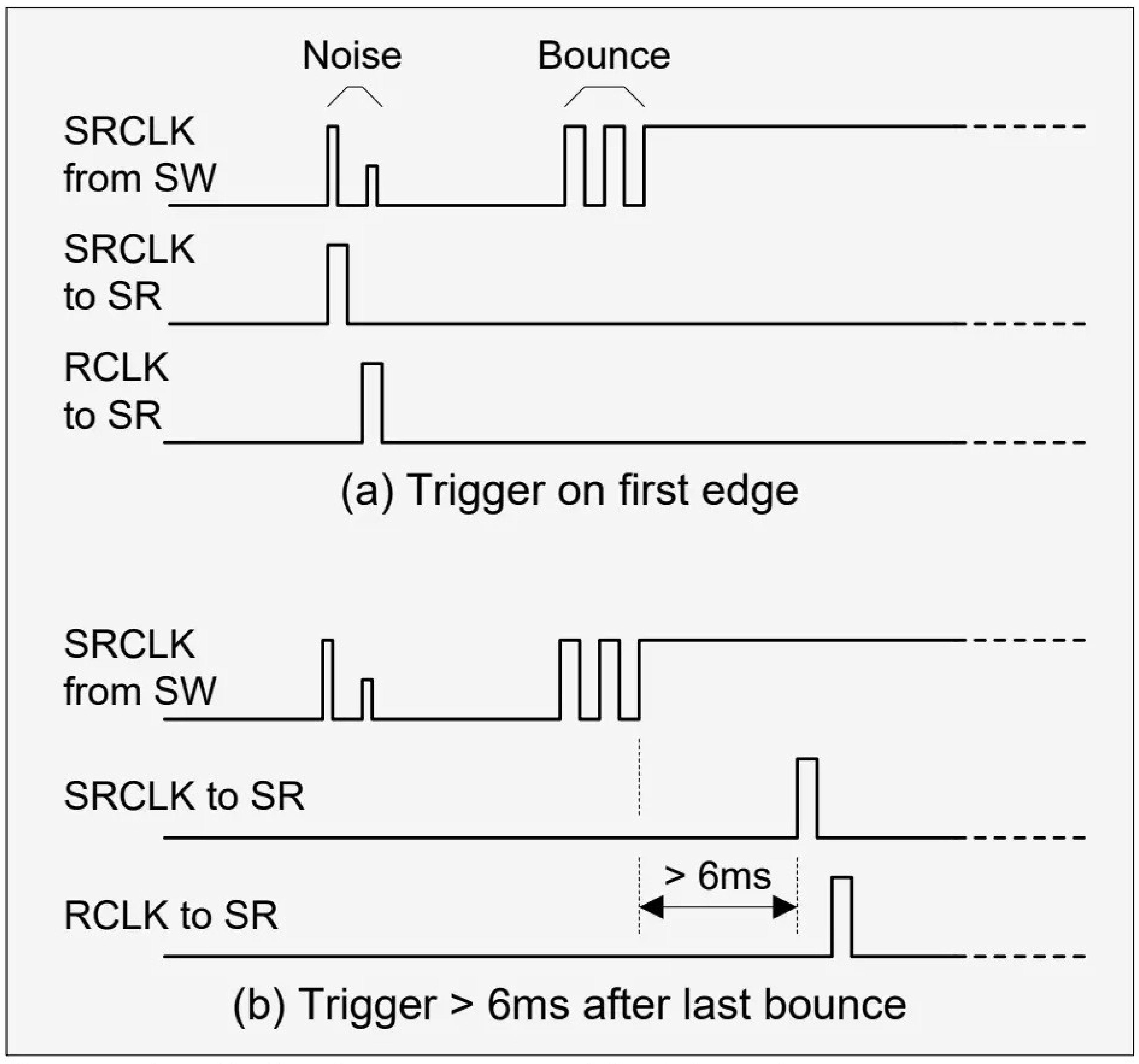


Fig.13: electrical noise can cause false triggering.

ideal world, starting from the switch being in its inactive state, the Arduino would detect the leading edge of the activation and cause its output to the shift register (SR) to respond immediately."

This is essentially how we implemented things, and how we depicted them in Fig.3. However, we also noted that, "… this approach may not be quite as ideal as we might suppose."

The first problem with our existing solution is that it doesn't account for noise in the form of electromagnetic interference (EMI). This can originate from natural sources, such as lightning, or human-made sources, including radio transmitters or motors switching on and off.

When our existing program is waiting for the **SRCLK** switch to be pressed, it reacts to the first active edge on the **SRCLK** signal. So, consider what would happen if noise were present on this signal, as shown in Fig.13(a).

Our current code would 'jump the gun', as it were. It would be, err, unfortunate if someone were using this code to control something like a missile launch!

To be honest, EMI is probably not something we need to worry about in the case of our current home-based experiments, but it is a significant concern in mission-critical and safety-critical applications such as industrial automation, medical devices, and aerospace.

One of the goals of these columns is to learn and apply 'best practices', enabling the knowledge gained here and now to be effectively deployed in real-world applications later.

It may be worth reminding ourselves that the timing relationships in the waveforms depicted in our illustrations, such as Fig. 13, are not to scale. For example, we might take a coffee break or go for a long walk between adjacent presses of our **SRCLK** switch, which means the noise could occur seconds, minutes, hours (even days) before the next switch press.

We're assuming that our switches can bounce anywhere from 1 to 100 times over a period not exceeding 6ms (six milliseconds).

Why did I pick 6ms when many textbooks quote only 1ms? Well, several years ago, one of my friends, an acknowledged expert in embedded system design, performed extensive tests on a wide variety of switches, finding a maximum bounce time on a big old switch from his collection to be around 6ms.

So, what we'll do, as illustrated in Fig.13(b), is wait until the signals from our switches have been steady for longer than 6ms following the last bounce before taking any action.

Let's pause here for just a moment. Does anything in the preceding paragraph strike you as odd? We have just stated that we will assume our switch bounce won't last longer than 6ms.

This implies that all we need to do is wait at least 6ms after the **initial transition** at the **beginning** of the bouncing sequence (ie, the first transition). Instead, we're now saying that we're going to wait for 6ms **after** the **final bounce**. Why?

Well, if we simply waited for 6ms **after the first transition**, we could still be fooled by any noise on the signal. Waiting for 6ms **after the final bounce** ensures that the signal has remained stable in its new state for this amount of time, thereby providing a high degree of confidence that we've just experienced a legitimate switching event.

How long should we wait?

The next consideration is that simply stating "wait for longer than 6ms" is somewhat vague. How much longer?

Well, the US Air Force has historically specified a 20ms debounce time after the last bounce for mechanical switches. This value was chosen based on empirical studies of various switch types conducted circa the 1940s and 1950s.

The desire was to ensure reliable operation under harsh conditions, such as vibration and mechanical shock. Once adopted by the Air Force, this value became a de facto standard in aerospace and military-grade electronics in the US, and many other military and aerospace standards worldwide have adopted the same rule.

In less critical deployments, such as those in industrial and consumer electronics, debounce times can be as low as 5ms or even 1ms, depending on the switch and the application. In the case of embedded systems and microcontroller applications, software-based debouncing typically employs a delay of 5–10ms, although some conservative designs still use a 20ms delay.

Now, you may be thinking that 20ms is a long time, but everything is relative (even my Auntie Barbara). In the case of an application like ours, where pressing a switch causes one or more LEDs to respond, most people won't perceive any delay in response of anything less than 50ms.

On the other hand, if we are talking about an automated system, such as a robotic assembly line or a pick-and-place machine, a 20ms delay between switch activation and response would be far too long, particularly in high-speed applications.

There are some cunning techniques we can use to reduce this response time to less than 1/10 of a millisecond, but they are outside the scope of these discussions. Still, perhaps we will discuss them some day, after we've completed the retro gaming system we are working on (in theory, at least)!

So, what we are going to do is wait for 16ms following the final bounce. Why 16ms? All will be revealed in our next column.

Homework assignment #2

Start thinking about how we can write the code to implement the type of switch debounce illustrated in Fig.13(b) with a delay of 16ms following the final bounce.

Note that we're explicitly saying a delay equal to 16ms, not greater than 16ms and certainly not less than 16ms.

Next time

I'll leave you with something to think about: how much easier having a microcontroller makes this debouncing logic. In the olden days, they had to do it with discrete components like resistors, capacitors and Schmitt-trigger gates. How times have changed!

When next we meet, we will commence by considering the solutions for all the "homework assignments" posed in this column.

After that… well, the world is our lobster. Until that happy day, if you have any thoughts that you'd care to share on anything you've read here, please feel free to drop me an email at max@clivemaxfield.com. And, as always, have a good one!

I have no idea why everyone calls me "Mad Max"!

Useful Bits and Pieces from Our Arduino Bootcamp series

Arduino Uno R3 microcontroller module	https://pemag.au/link/ac2g
Solderless breadboard	https://amzn.to/3O2L3e8
8-inch (20cm) jumper wires (male-to-male)	https://amzn.to/3O4hnxk
Long-tailed 0.1-inch (2.54mm) pitch header pins	https://pemag.au/link/ac2h
LEDs (assorted colours)	https://amzn.to/3E7VAQE
Resistors (assorted values)	https://amzn.to/3O4RvBt
Ceramic capacitors (assorted values)	https://pemag.au/link/ac2i
16V 100µF electrolytic capacitors	https://pemag.au/link/ac2j
Momentary pushbutton switches	https://amzn.to/3Tk7Q87
Kit of popular SN74LS00 chips	https://pemag.au/link/ac2k
74HC595 8-bit shift registers	https://pemag.au/link/ac1n

Other stuff

Soldering guide	https://pemag.au/link/ac2d
Basic multimeter	https://pemag.au/link/ac2f

Components for Weird & Wonderful Projects, part 1

4-inch (10cm) jumper wires (optional)	https://pemag.au/link/ac2l
8-segment DIP red LED bar graph displays	https://pemag.au/link/ac2c

Components for Weird & Wonderful Projects, part 2

144 tricolour LED strip **(required)**	https://pemag.au/link/ac2s
Pushbuttons (assorted colours) **(required)**	https://pemag.au/link/ac2t
7-Segment display modules **(recommended)**	https://pemag.au/link/ac2u
2.1mm panel-mount barrel socket **(optional)**	https://pemag.au/link/ac2v
2.1mm inline barrel plug **(optional)**	https://pemag.au/link/ac2w
25-way D-sub panel-mount socket **(optional)**	https://pemag.au/link/ac2x
25-way D-sub PCB-mount plug* **(optional)**	https://pemag.au/link/ac2y
15-way D-sub panel-mount socket **(optional)**	https://pemag.au/link/ac2z
15-way D-sub PCB-mount plug^ **(optional)**	https://pemag.au/link/ac30

* one required for each game cartridge you build
^ one required for each auxiliary control panel you build

Components for Weird & Wonderful Projects, part 3

22 AWG multicore wire kit	https://pemag.au/link/ac3m
20 AWG multicore wire kit	https://pemag.au/link/ac3n

Components for Weird & Wonderful Projects, part 4

5V Buck converter module	https://pemag.au/link/ac2m
9V 2A power supply **(optional)**	https://pemag.au/link/ac46
Bench power supply **(optional)**	https://pemag.au/link/ac48

DOWNLOAD YOUR PE/EPE BACK ISSUES

COMPLETE-YEAR collections – all 12 issues for each year!

From just £5.95 each

Volume	Period	Price
VOL 28	January to December 1999	**£5.95**
VOL 29	January to December 2000	**£6.95**
VOL 30	January to December 2001	**£7.95**
VOL 31	January to December 2002	**£7.95**
VOL 32	January to December 2003	**£7.95**
VOL 33	January to December 2004	**£7.95**
VOL 34	January to December 2005	**£8.95**
VOL 35	January to December 2006	**£8.95**
VOL 36	January to December 2007	**£8.95**
VOL 37	January to December 2008	**£8.95**
VOL 38	January to December 2009	**£10.95**
VOL 39	January to December 2010	**£11.95**
VOL 40	January to December 2011	**£12.95**
VOL 41	January to December 2012	**£14.95**
VOL 42	January to December 2013	**£15.95**
VOL 43	January to December 2014	**£16.95**
VOL 44	January to December 2015	**£17.95**
VOL 45	January to December 2016	**£19.95**
VOL 46	January to December 2017	**£22.95**
VOL 47	January to December 2018	**£27.95**
VOL 48	January to December 2019	**£27.95**
VOL 49	January to December 2020	**£29.95**
VOL 50	January to December 2021	**£32.95**
VOL 51	January to December 2022	**£34.95**
VOL 52	January to December 2023	**£34.95 – NEW!**
VOL 53	January to December 2024	**£34.95 – NEW!**

FIVE-YEAR collections – 60 issues!

From just £14.95 each

Code	Period	Price
PE2832:	**Jan 99 to Dec 03**	**£14.95**
PE2933:	**Jan 00 to Dec 04**	**£14.95**
PE3034:	**Jan 01 to Dec 05**	**£14.95**
PE3135:	**Jan 02 to Dec 06**	**£14.95**
PE3236:	**Jan 03 to Dec 07**	**£14.95**
PE3337:	**Jan 04 to Dec 08**	**£14.95**
PE3438:	**Jan 05 to Dec 09**	**£19.95**
PE3539:	**Jan 06 to Dec 10**	**£25.95**
PE3640:	**Jan 07 to Dec 11**	**£25.95**
PE3741:	**Jan 08 to Dec 12**	**£25.95**
PE3842:	**Jan 09 to Dec 13**	**£33.95**
PE3943:	**Jan 10 to Dec 14**	**£33.95**
PE4044:	**Jan 11 to Dec 15**	**£33.95**
PE4145:	**Jan 12 to Dec 16**	**£35.95**
PE4246:	**Jan 13 to Dec 17**	**£37.95**
PE4347:	**Jan 14 to Dec 18**	**£38.95**
PE4448:	**Jan 15 to Dec 19**	**£39.95**
PE4549:	**Jan 16 to Dec 20**	**£42.95**
PE4650:	**Jan 17 to Dec 21**	**£44.95**
PE4751:	**Jan 18 to Dec 22**	**£44.95 – NEW!**
PE4852:	**Jan 19 to Dec 23**	**£44.95 – NEW!**

20 YEARS OF PE!

AMAZING BUNDLE OFFER
Every issue from 1999 – 2018

240 magazines for just

£69.95

PE2832: Jan 99 to Dec 03
PLUS!
PE3337: Jan 04 to Dec 08
PLUS!
PE3842: Jan 09 to Dec 13
PLUS!
PE3447: Jan 14 to Dec 18

Download your collections at: www.electronpublishing.com

180-230V DC Moto

Power
0-230V DC
Output
SILICON CHIP
180V DC MOTOR CONTROLLER
0
10
20
30
40
50
60
70
80
100
SPEED
Reset
Run
WARNING!
This Speed Controller circuit operates directly from the 230VAC mains supply and contact with live components and exposed connections can be lethal, unless disconnected from the mains.

- Controls 180-230V DC motors rated from 1A to 10A (¼HP to 2.5HP)
- Controlled by four common op amp ICs with one opto-coupler and three linear regulators
- Zero to full speed control
- Safe startup procedure
- Emergency cut-out switch facility
- Automatic over-current switch-off
- Optional reversing switch capability
- PWM, Live and Power indicator LEDs
- Rugged diecast aluminium enclosure
- Current and back-EMF monitoring for speed regulation under load
- Initial setup adjustments can be made with a low-voltage supply

DC motors from 180-230V DC power various pieces of equipment; they are particularly common in treadmills. Often, these motors are removed from the treadmill, possibly because the speed controller has failed.

Such motors can be reused for other purposes, such as adding computer control to a lathe. Many of these motors are sold via the internet on sites such as eBay, often inexpensively.

The type of DC motor we are referring to here typically has permanent magnets in the stator and field coils for the rotor. The electrical connection to the field coils is made via a commutator and brushes.

These motors can be powered from the mains using a full-wave bridge rectifier to convert the 230 AC voltage from the mains to a pulsating DC voltage, where the voltage rises and falls in a sinewave shape over each half of the mains waveform. The resulting average voltage is close to 230V DC.

Warning: Mains Voltage

This Speed Controller operates directly from the 230V AC mains supply; contact with any live component is potentially lethal. Do not build it unless you are experienced working with mains voltages.

Scope 1 shows the resulting 'DC' waveform that is applied to the motor for it to run at full speed. If you want to slow the motor down, you need a speed controller.

Our DC Motor Speed Controller provides the same 100Hz pulsating DC mains rectified voltage as a full-wave rectifier, but it adds speed control by switching this waveform on and off more rapidly, at around 900Hz. The averaged DC voltage is thus multiplied by the proportion of time it is switched on.

This type of drive is called pulse-width modulation (PWM), where voltage is applied to the load in a series of pulses with a duty cycle (0-100%) that can be adjusted to control the average voltage applied to the motor.

With the duty cycle set at 100%, the motor is driven with the full 100Hz full-wave rectified mains voltage. As the duty cycle reduces, so does the average voltage applied to the motor. A 50% duty cycle reduces the average voltage by one-half.

Scope 2 shows the PWM-chopped full-wave rectified mains with it on about 60% of the time (a 60% duty cycle). The resulting averaged voltage should be about 60% of the full waveform shown in Scope 1. The mean measured value of 113V is 58% of the 195V reading from Scope 1.

Note that some mains motor speed controllers designed to work with universal motors will be able to control the speed of DC motors, especially if they rectify the incoming mains immediately. However, such controllers are usually optimised for use with AC motors, so they lack some features that are desirable for use with DC motors.

Our new design

By designing a controller purely for DC motors, we can provide the best control of this type of motor. For example, a typical universal motor speed controller might have feedback to maintain motor speed under load, but only by monitoring the motor current. That type of feedback control does not monitor motor speed and relies on the motor current being indicative of motor speed and load.

One problem with that is that, under load, the motor speed drops and it draws more current. This extra current causes the controller to increase the PWM duty cycle to increase the motor speed. That increases the current further, so the controller increases the duty cycle further. The process can quickly become unstable, possibly producing bursts of voltage to the motor. That is especially likely to happen if too much motor speed correction is applied.

r Speed Controller

High-voltage DC motors are commonly used in lathes, consumer-grade treadmills, industrial conveyor belts and similar equipment. This Speed Controller can control such a motor over a wide range of speeds, from very slow to full speed. A constant speed is maintained even with a varying load due to motor-generator voltage (back-EMF) sensing and current feedback circuitry.

Part 1 by John Clarke

In this new design, we also monitor the motor speed using back-EMF (electromotive force). The motor generates this back-EMF from its rotation and it directly indicates the motor speed – see Scope 3. The generated voltage will drop when the motor is loaded, since it will slow down.

Increasing the duty cycle of voltage applied to the motor can compensate for this reduction in motor speed. This is a negative feedback control, so it is far more stable than feedback based purely on motor current.

Compare Scope 3, with the motor driven by switched PWM, to Scope 2. You can see that in Scope 3, the motor load also generates a voltage (back-EMF) during the PWM off-times. This is seen as the spikes below the baseline indicated by the small arrow and “1” on the left. The motor is generating spikes of around -85V.

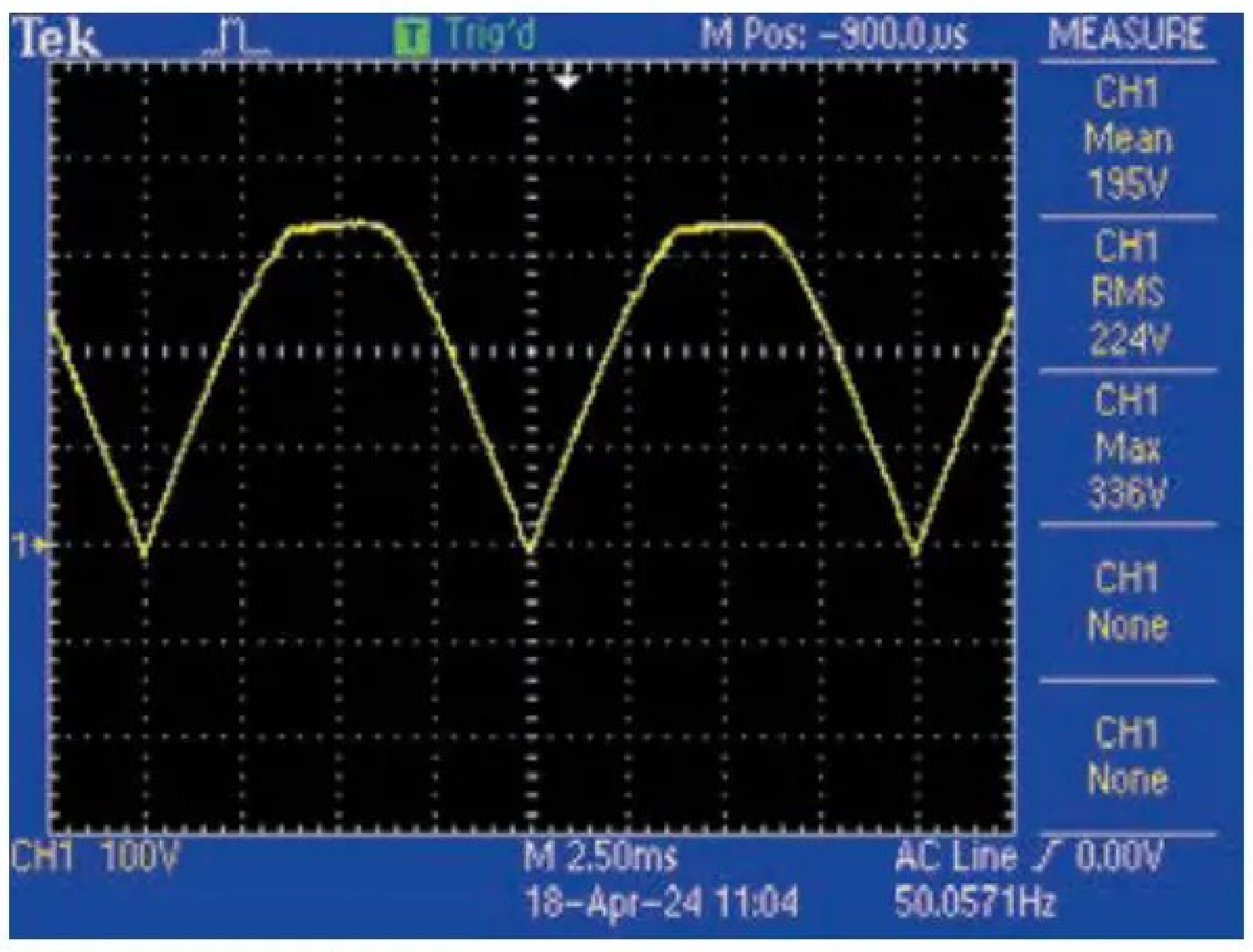
Applying a combination of both current and back-EMF feedback control, with a measured amount of each, provides excellent speed control without the risk of speed instability.

Incidentally, we don’t use back-EMF speed control for a universal motor controller because that voltage is essentially non-existent. The generated voltage relies on having a magnetised core. Since the universal motor has windings for both the armature and stator, there is little remnant magnetism when the windings are not powered. Hence, little to no voltage is produced.

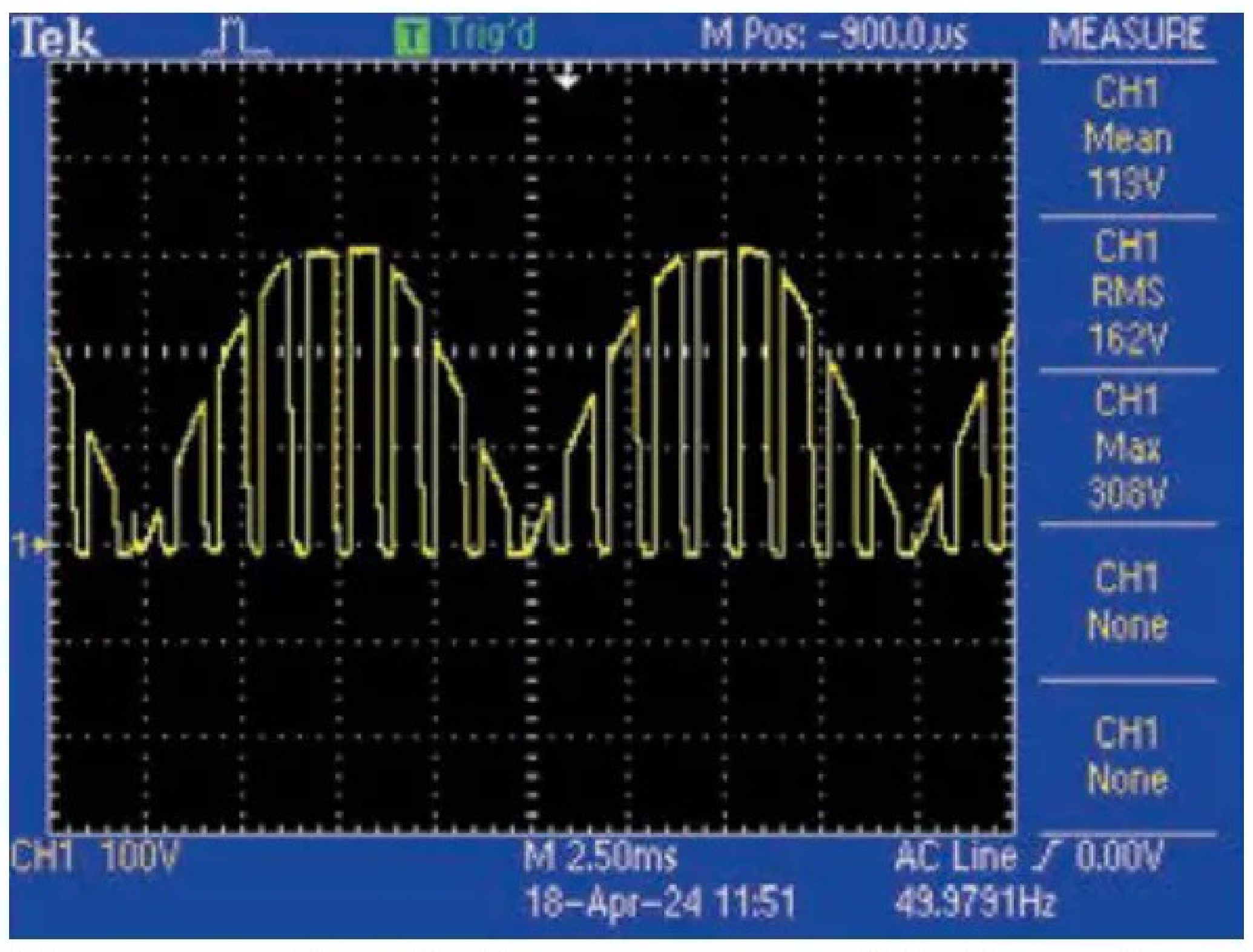
However, with a DC motor, the stator can comprise permanent magnets or powered windings, so a voltage is induced in the rotor as it spins.

Design inspiration

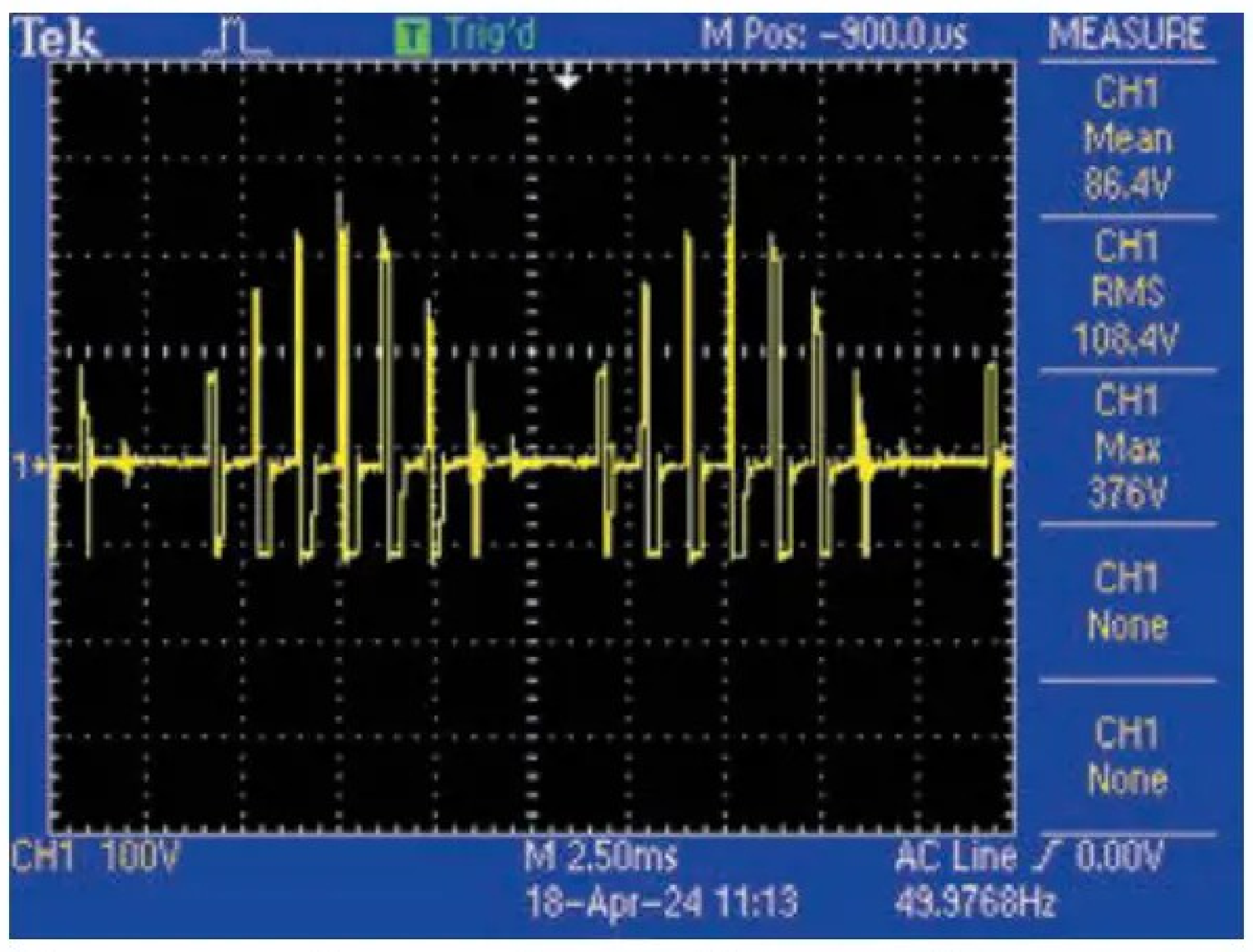
We based our new controller on the features of a small lathe controller (the Sieg C1 micro lathe) that was



Scope 1: this is the pulsating ‘DC’ waveform that is applied to the motor for it to run at full speed, created by rectifying the 230V AC mains. Note that the mean (average) voltage is lower than the RMS. This waveform (like the others) was captured using an isolated differential probe.



Scope 2: the full-wave rectified mains being switched on and off with a duty cycle of about 60% (60% on, 40% off). As expected, the resulting average voltage is about 60% of the full waveform in Scope 1.



Scope 3: the motor voltage when driven by the switched PWM version of the full-wave rectified mains waveform. You can see how the motor load generates negative voltage spikes of about -85V (back-EMF) during the PWM off periods while the motor is not being driven.

extensively documented by Dr Hugo Holden. He provides considerable detail on the operation of that circuit, as well as a description of how it uses op amps, in a PDF on his website at https://pemag.au/link/abmn

While the Sieg controller is for small motors rated up to 1A, we have designed ours to handle any high-voltage DC motor from 1A to 10A.

The Sieg motor controller includes several safety features we also incorporated into our new design. One such feature is the facility for a safety/emergency stop switch, where the motor does not start when the switch is open and will shut off if it opens during operation. This feature does not need to be used; it can be bypassed with a wire loop if not required.

However, it's a great idea to include such a switch for a lathe or any other piece of industrial equipment. A lathe can be set up so it will not run unless a safety shield is in place to protect against flying debris. In other cases, the safety switch can be a large, red emergency stop button (available from Farnell). You could even have both simply by wiring them in series. Whenever one opens, the motor will stop.

There are other conditions under which the motor won't run for safety. For example, when the controller is initially powered up or if the motor becomes overloaded during operation. The motor cannot start or be restarted unless the speed control is brought back to the full anti-clockwise position before being advanced to start the motor running. That assumes the safety switch is closed and there is no motor overload.

The Sieg controller does this with a switch incorporated within the speed potentiometer. That is not terribly unusual as many vintage radios and some music instrument amplifiers have a power switch within the volume potentiometer, where rotating the potentiometer fully anticlockwise opens the switch to disconnect the power.

However, in the Sieg controller and our design, the switch must be closed when the potentiometer is fully anticlockwise and open when it is advanced clockwise. That is the reverse of a radio/amplifier potentiometer, making it a rather unique and difficult part to obtain.

Even if we decided to use that style of potentiometer, with (say) a relay to reverse the switching sense, it still would not be ideal. That's because volume control potentiometers have a logarithmic resistance change over their rotation, while we need a linear response.

Therefore, we use a standard potentiometer that does not have a switch incorporated and instead provide the switching feature using a comparator and relay. That avoids having to source a special type of potentiometer. The comparator monitors the potentiometer's wiper voltage and switches a relay according to the potentiometer position.

How it works

The basic block diagram is shown in Fig.1. The circuitry is based around a full-wave bridge rectifier that provides the 100Hz half-sinewave 'DC' voltage from the mains. The IGBT that switches power on and off to the motor is connected in series with the motor, between the DC terminals of the bridge rectifier.

The IGBT's gate is driven by PWM circuitry that controls the motor speed. The duty cycle of the PWM signal depends on the position of the speed potentiometer (VR1), the motor current (detected by sense resistors) and the motor with both back-EMF from its negative terminal.

The PWM signal is generated by comparing voltage V_O (derived from the speed potentiometer voltage and the current and voltage feedback) to a sawtooth waveform. V_O comes from the output of op amp IC2c, while

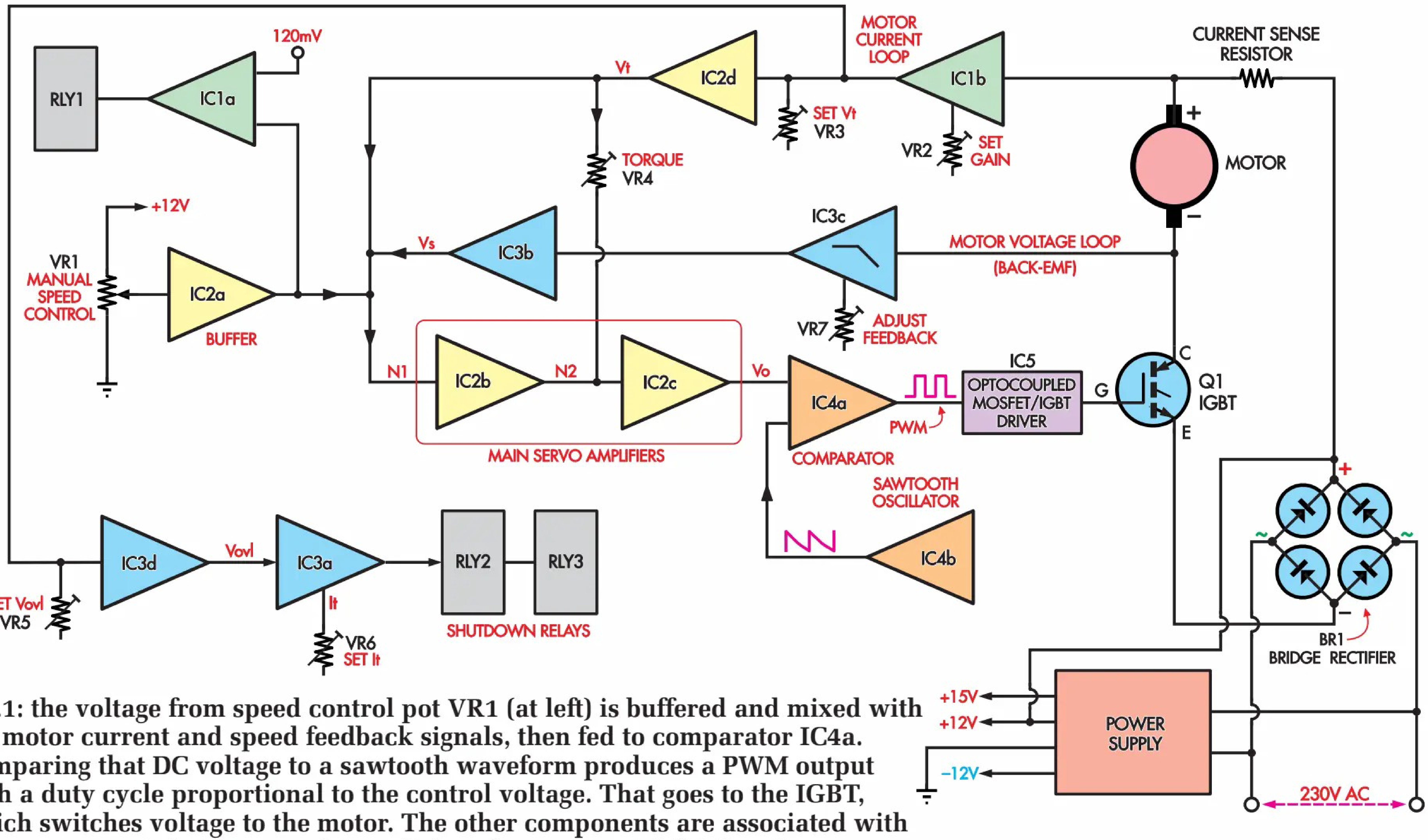

RLY1
IC1a
120mV
Vt
IC2d
MOTOR CURRENT LOOP
IC1b
CURRENT SENSE RESISTOR
SET Vt
VR3
VR2
SET GAIN
+
MOTOR
−
TORQUE
VR4
+12V
VR1
MANUAL SPEED CONTROL
IC2a
BUFFER
Vs
IC3b
IC3c
MOTOR VOLTAGE LOOP
(BACK-EMF)
VR7
ADJUST FEEDBACK
N1
IC2b
N2
IC2c
Vo
IC4a
PWM
COMPARATOR
IC5
OPTOCOUPLED MOSFET/IGBT DRIVER
G
C
E
Q1
IGBT
MAIN SERVO AMPLIFIERS
SAWTOOTH OSCILLATOR
IC4b
+
~
~
−
IC3d
Vovl
IC3a
It
RLY2
RLY3
SET Vovl
VR5
VR6
SET It
SHUTDOWN RELAYS
BR1
BRIDGE RECTIFIER
+15V
+12V
−12V
POWER SUPPLY
230V AC

Fig.1: the voltage from speed control pot VR1 (at left) is buffered and mixed with the motor current and speed feedback signals, then fed to comparator IC4a. Comparing that DC voltage to a sawtooth waveform produces a PWM output with a duty cycle proportional to the control voltage. That goes to the IGBT, which switches voltage to the motor. The other components are associated with the over-current shutdown, safety switch and power-up inhibit functions.

the sawtooth waveform is from the oscillator based on IC4b. IC4a compares the sawtooth waveform to V_O and generates the variable duty-cycle PWM output.

The manual speed control voltage from potentiometer VR1 is buffered via IC2a, which applies a DC voltage to the N1 node, at the input to IC2b. Voltages from the feedback loop outputs, from IC3b and IC2d, are also applied to N1. These provide the speed (V_S) and torque (V_t) signals, which are derived from the motor back-EMF and the motor current, respectively.

For torque feedback, the motor current is determined by the voltage across the current sense resistance in series with the motor's positive terminal. The resulting voltage is amplified by IC1b, which has its gain set by trimpot VR2. The gain is adjusted according to the motor current rating; more gain is used for low-current motors and less for higher-current motors.

The resulting voltage is applied to op amp IC2d, with an offset voltage set via trimpot VR3.

The motor's back-EMF voltage is amplified by IC3c and is offset using trimpot VR7 before being applied to op amp IC3b. IC3c provides a low-pass filter with a roll-off point of 1.6Hz to remove the 100Hz ripple.

The torque voltage from IC2d (V_t) and the speed signal from IC3b (V_S) are summed with the speed control signal at N1, while a separate torque signal is applied to the N2 junction of the main servo amplifiers, IC2b and IC2c, via VR4. This provides feedback adjustment, allowing for the best speed control as the motor is loaded.

For current overload detection, the motor current is amplified by op amp IC3d and compared against a threshold by comparator IC3a (it's actually an op amp but used in open-loop mode, so it acts as a comparator). If the current limit is exceeded, the motor is switched off by the shutdown relay, RLY2, which switches off RLY3. That disconnects power from the motor. These relays return to their normal positions when the overload is cleared and the speed control is rotated fully anti-clockwise.

VOLTS
DC 220V
RPM
4000
INSUL
F
AMB
40°C
07112072

This is the motor we used for testing, You can find lots of similar second-hand and new motors on eBay.

IC1a detects when the speed control pot is set at zero. It compares the buffered speed control voltage from IC2a against a 119mV reference. The Relay (RLY1) is powered, opening the connected contacts, when the voltage from the speed control is higher than 119mV. The open relay contacts prevent the motor from starting, although it will continue to run if it is already running.

Other relays are connected in series with the coil of the one that powers the motor, so if any of them open, that will stop the motor from spinning. That includes if an overload is detected or the emergency/safety switch is open. For the motor to run, the speed potentiometer must be set almost fully anti-clockwise (switching RLY1 off), then rotated more clockwise to start the motor.

The circuit has three main supply rails: a ±12V split supply and a +15V supply. The +12V supply connects to the positive output of the full-wave bridge rectifier, so the whole circuit operates at mains potential. Level shifting between IC4a's output and the gate of IGBT Q1 is via an opto-coupled Mosfet/IGBT gate driver (IC5).

Circuit details

The full circuit is shown in Fig.2. It comprises five ICs, an IGBT, several diodes, two transistors, three relays plus numerous capacitors and resistors.

A significant component for PWM drive generation is op amp IC4b, wired as a sawtooth oscillator (near the bottom of the diagram). This is the second amplifier in dual LM833 op amp IC4. It is powered from the ±12V supply.

A bias voltage is generated using two 100kΩ resistors connected in series across the ±12V supply. The centre connection of this voltage divider is connected to the non-inverting input, pin 5 of IC4b. The voltage at pin 5 would be 0V, except for the fact that there is also a 47kΩ resistor between pin 5 and the op amp output, pin 7.

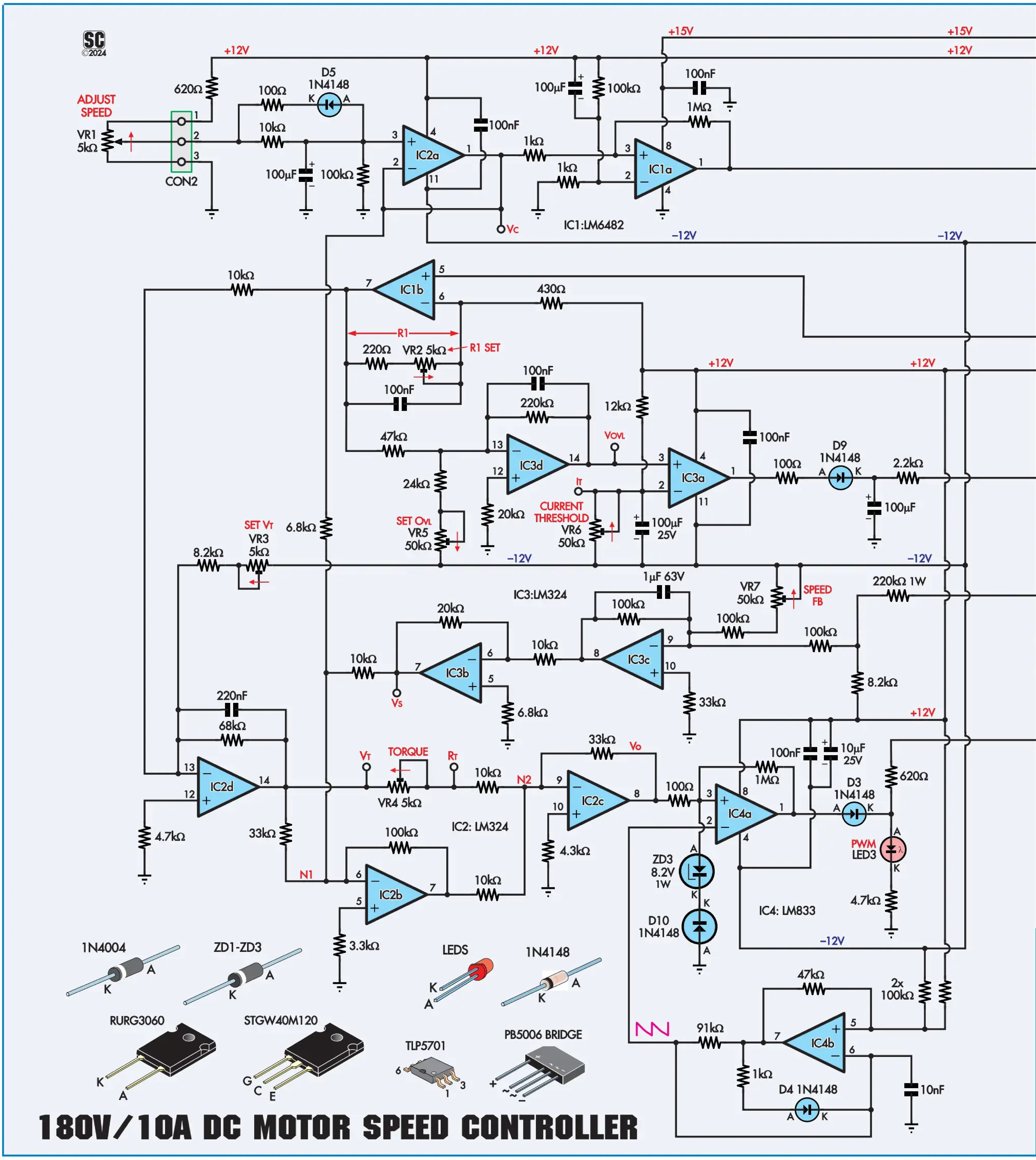
To calculate the voltage at pin 5, the two 100kΩ resistors connected across the ±12V supply can be considered a 50kΩ resistor between pin 5 and 0V. That leaves a 47kΩ/50kΩ voltage divider, with the voltage at the end of the 47kΩ resistor shifting between about 10.9V when pin 7 is high and -10.9V when it is low.

So when the op amp output is high, the 47kΩ resistor pulls pin 5 to around 5.6V, and when the op amp output is low, it is around -5.6V. The output oscillates between the high and low states due to the 10nF capacitor at the inverting input (pin 6) and the charge and discharge resistances between pins 6 and 7.

When power is first applied, the 10nF capacitor is discharged, so pin 6 is near -10.9V. Pin 5 is at a higher voltage than that, so pin 7 goes high. Pin 5 then sits at around 5.6V. The 10nF capacitor charges via the 1kΩ resistor and diode D4. As soon as the capacitor at pin 6 charges just beyond the pin 5 voltage, pin 7 goes low, to -10.9V, since the inverting input voltage is above the non-inverting input.

Pin 5 is then at -5.6V, and the capacitor discharges via the 91kΩ resistor. Diode D4 is reverse-biased and does not take part in the discharge cycle. Once the capacitor voltage exceeds -5.6V, the pin 7 output goes high again, and the process repeats.

So the 10nF capacitor is charged quickly via the 1kΩ resistor and


SC ©2024
ADJUST SPEED
VR1 5kΩ
CON2
620Ω
+12V
100Ω
D5 1N4148
10kΩ
100µF
100kΩ
IC2a
100nF
Vc
1kΩ
100µF
100kΩ
+15V
100nF
1MΩ
IC1a
IC1:LM6482
–12V
10kΩ
IC1b
430Ω
R1
220Ω
VR2 5kΩ
R1 SET
100nF
47kΩ
220kΩ
IC3d
24kΩ
SET OVL
VR5 50kΩ
20kΩ
12kΩ
VOVL
IT
CURRENT THRESHOLD
VR6 50kΩ
100µF 25V
IC3a
100nF
D9 1N4148
2.2kΩ
100µF
SET VT
VR3 5kΩ
8.2kΩ
6.8kΩ
IC3:LM324
1µF 63V
100kΩ
VR7 50kΩ
SPEED FB
220kΩ 1W
20kΩ
10kΩ
IC3b
Vs
6.8kΩ
IC3c
33kΩ
8.2kΩ
220nF
68kΩ
IC2d
4.7kΩ
33kΩ
N1
VT
TORQUE
RT
VR4 5kΩ
N2
33kΩ
Vo
IC2c
4.3kΩ
IC2: LM324
IC2b
3.3kΩ
100Ω
IC4a
100nF
10µF 25V
1MΩ
D3 1N4148
620Ω
PWM LED3
4.7kΩ
ZD3 8.2V 1W
D10 1N4148
IC4: LM833
47kΩ
2x 100kΩ
91kΩ
IC4b
1kΩ
D4 1N4148
10nF
1N4004
ZD1-ZD3
LEDS
1N4148
RURG3060
STGW40M120
TLP5701
PB5006 BRIDGE
A
K
G
C
E

180V/10A DC MOTOR SPEED CONTROLLER

diode D4, then discharged much more slowly via the 91kΩ resistor. The resulting waveform is described as a sawtooth shape, rising quickly and falling more slowly. The waveform ranges from about +5.6V to -5.6V at about 900Hz.

The sawtooth waveform at pin 6 of IC4b goes to the inverting (pin 2) input of IC4a, which compares it against the V_O voltage from IC2c (at its non-inverting pin 3 input). The output of IC4a goes high (to around 10.9V) when the voltage from the IC4a oscillator is lower than the feedback voltage; its output is low (-10.9V) otherwise.

IC4a's output is therefore a rectangular waveform with a higher duty cycle (higher for longer) when the V_O voltage is higher. IC4a's output drives optically-coupled Mosfet/IGBT driver

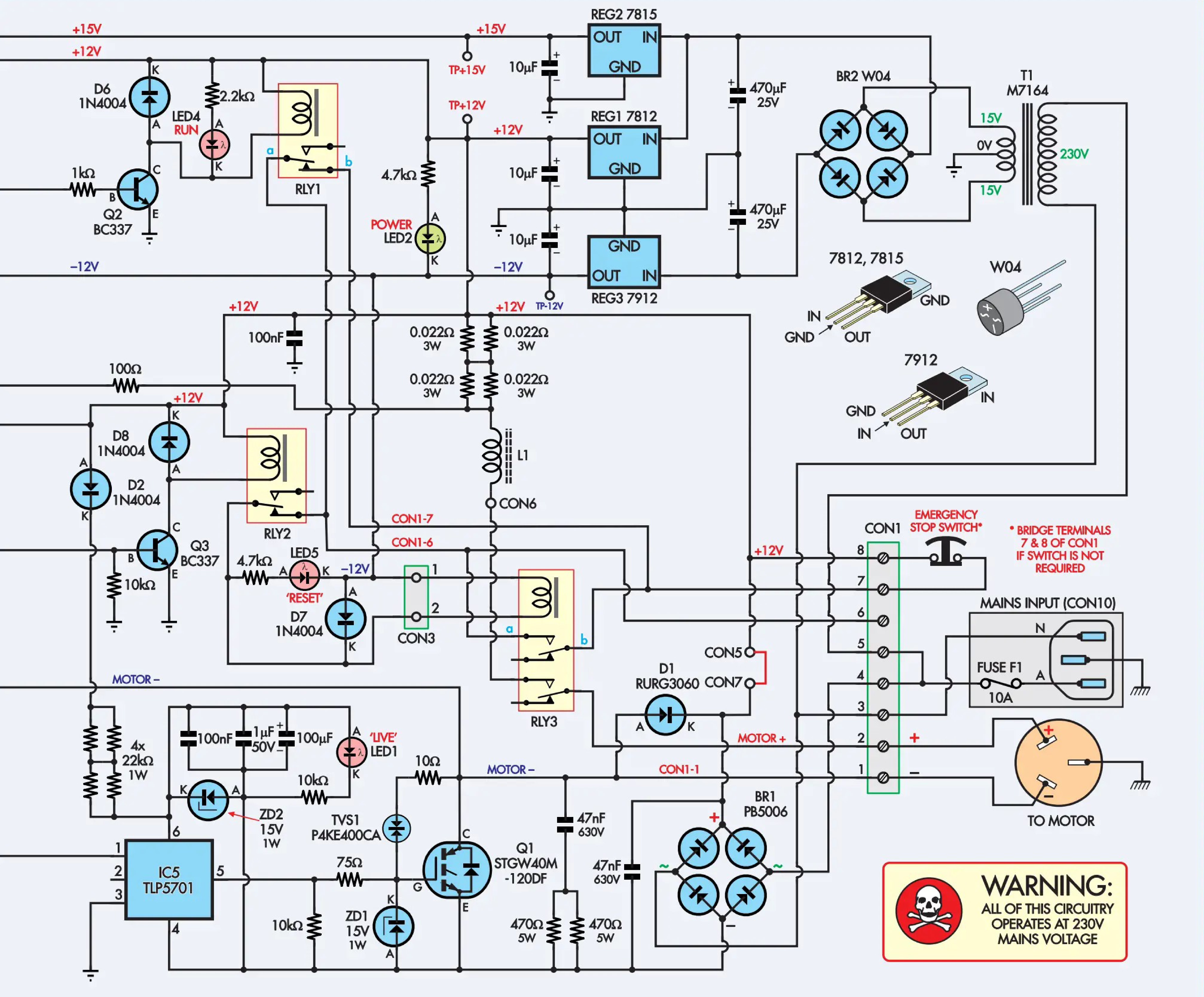

+15V
+12V
D6 1N4004
2.2kΩ
LED4 RUN
RLY1
1kΩ
Q2 BC337
–12V
REG2 7815
OUT IN GND
TP+15V
TP+12V
10μF
REG1 7812
REG3 7912
4.7kΩ
POWER LED2
TP-12V
470μF 25V
BR2 W04
T1 M7164
15V
0V
230V
7812, 7815
IN GND OUT
W04
7912
100nF
0.022Ω 3W
100Ω
D8 1N4004
D2 1N4004
Q3 BC337
10kΩ
RLY2
4.7kΩ
LED5 'RESET'
D7 1N4004
CON3
L1
CON6
CON1-7
CON1-6
RLY3
CON1
EMERGENCY STOP SWITCH*
* BRIDGE TERMINALS 7 & 8 OF CON1 IF SWITCH IS NOT REQUIRED
MAINS INPUT (CON10)
FUSE F1 10A
N
A
E
CON5
CON7
D1 RURG3060
MOTOR +
MOTOR –
CON1-1
TO MOTOR
4x 22kΩ 1W
100nF
1μF 50V
100μF
'LIVE' LED1
10Ω
10kΩ
ZD2 15V 1W
TVS1 P4KE400CA
IC5 TLP5701
75Ω
10kΩ
ZD1 15V 1W
Q1 STGW40M -120DF
47nF 630V
470Ω 5W
BR1 PB5006
WARNING:
ALL OF THIS CIRCUITRY OPERATES AT 230V MAINS VOLTAGE

Fig.2: virtually all the op amp based control circuitry is on the left half of the diagram; 12 op amp stages are used, inside four ICs, two of which operate as comparators. The IGBT and its driving circuitry are at lower middle, with the current measurement shunts and relays that switch power to the motor above that. The linear power supply is at upper right.

IC5. It level-shifts IC4a's square-wave output to a voltage suitable for driving the gate of Q1.

Scopes 4 & 5 show the sawtooth waveform and Q1's PWM gate-drive signal at duty cycles of about 10% and 90%. The top trace (yellow) is the PWM signal, while the lower cyan trace is the sawtooth waveform, which oscillates between ±5.6V. The yellow horizontal dotted line (representing V_O) shows how the PWM output is high when the sawtooth waveform is below that level.

Scope 6 shows the PWM drive to Q1's gate. The rise and fall times of the waveform are 1.37μs and 1.2μs, respectively, with the gate voltage reaching 14.8V. We want the switching time to be short to minimise heating in Q1 during partial conduction periods, and the gate drive needs to be within its ±25V rating while being high enough to fully switch it on, which occurs at around 15V.

The 15V supply for driving the gate comes from the positive output of the bridge rectifier via diode D2 and four series/parallel 22kΩ 1W resistors. This provides an average of 5mA to IC5 and the 15V zener diode (ZD2). That supply is smoothed to a DC voltage by a 100μF capacitor with parallel 1μF and 100nF ceramic capacitors so that IC5 can receive bursts of current for driving Q1's gate when needed.

LED1 is also driven by this 15V supply, so it indicates when power is being fed to the circuit from the positive side of the bridge rectifier. When lit, the entire circuit is at mains potential.

IC5 has an internal LED between pins 1 and 3, and its output goes high when that LED is powered via the 620Ω resistor from the output of IC4a and diode D3.

That diode prevents a negative voltage from being applied to IC5's internal LED when IC4a's output goes negative. The 620Ω series resistor limits the LED current to around 10mA. Q1's gate is driven via a 75Ω rate-limiting gate resistor.

The IGBT is protected from overvoltage at switch-off by transient voltage suppressor TVS1, which conducts if Q1's collector goes above 400V, causing the gate to be pulled high. That switches on Q1 to shunt the excess voltage. The 10Ω series resistor limits the current into protective 15V zener diode ZD1, which prevents the gate voltage from going beyond Q1's maximum limits.

Apart from TVS1, any high voltage spikes are also coupled via diode D1 into the 47nF capacitor connected across the + and – terminals of BR1. The capacitor absorbs some voltage transients.

A snubber between the IGBT's collector and emitter terminals prevents switch-off oscillations. It comprises a 47nF capacitor and two paralleled 470Ω 5W resistors.

Inductor L1, in series with the motor, limits the current rise rate when the motor is switched on to protect the IGBT from excessive surge current. It also helps to filter the 900Hz switching drive for the motor, reducing electromagnetic interference (EMI).

The motor current is monitored via a set of 0.022Ω shunt resistors. The four 0.022Ω 3W resistors are connected in series/parallel, yielding a 0.022Ω 12W resistance. They connect between the bridge rectifier positive terminal and L1. This shunt produces a voltage at the lower end that is below the +12V supply, in proportion to the current flow.

The motor voltage is monitored via a 220Ω 1W resistor from the motor's negative terminal. This voltage is also negative with respect to the +12V supply and becomes more negative with increased back-EMF voltage. The circuitry for monitoring these feedback voltages will be described separately.

Current feedback

IC1b amplifies the voltage across the current measurement shunt resistors. Its gain can be adjusted between 1.5 times (with trimpot VR2 at minimum resistance) and 13.1 times, when the resistance between pins 6 and 7 of IC1b is 5.22kΩ. This allows the circuit to work with motors rated between 1A and 10A with the correct overload threshold.

Note how IC1 has a 15V positive supply rather than 12V. This means the positive supply for IC1 is 3V above the 12V that the shunt resistor is referenced to. That way, the op amp output can reach 12V when there is no voltage across the shunt.

Even though the op amp is a rail-to-rail type, where the input and output voltages can be up to the supply rails, there will be some differences due to the input offset voltage of the op amp and the fact that the output can only reach within a few millivolts of its supply rails. So, with IC1's positive supply above 12V, the op amp has the headroom to handle 12V signals.

The output voltage from IC1b is amplified and inverted by op amp IC2d, which has a gain of -6.8, determined by the 10kΩ input resistor and the 68kΩ feedback resistor. This amplifier also acts as an integrator, filtering out the PWM signal and the pulsating DC due to the 220nF capacitor connected across the 68kΩ resistor.

The resulting low-pass filter has a roll-off point at about 10.6Hz, well below the 100Hz of the rectified mains and the 900Hz PWM signal. IC2d's output is applied to the N1 node via a 33kΩ resistor.

By the way, all inverting amplifiers in the motor controller circuit include a resistor from the non-inverting input of the op amp to the 0V rail. These are to equalise the two input impedances so that any input currents will balance out. This minimises offset voltages due to the parasitic input currents.

Note how op amp IC2d also connects to the -12V supply via an 8.2kΩ resistor and trimpot VR3. This preset trimpot adjusts the voltage offset at the V_T test point (IC2d's output). It is adjusted to provide the correct operation of the PWM by keeping voltage within range of the voltage swing from the sawtooth oscillator, IC4b.

The output voltage of IC1b is also applied to the current overload circuitry that comprises IC3d and IC3a; more on that later.

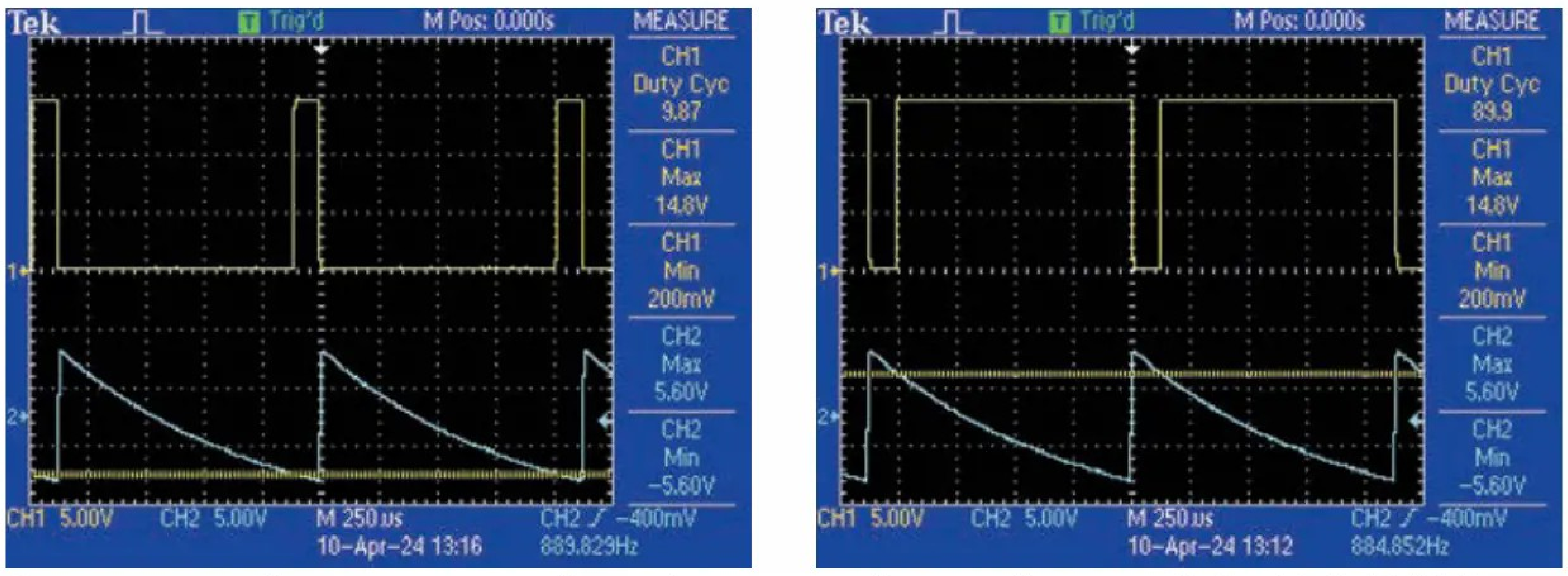
Motor speed feedback

The motor back-EMF voltage is monitored via a 220kΩ 1W resistor from the motor's negative terminal. This voltage is attenuated and offset toward the +12V rail by the connected 8.2kΩ resistor. The resulting voltage is applied to an inverting and integrating buffer, IC3c, via a 100kΩ resistor to its inverting input, pin 9.

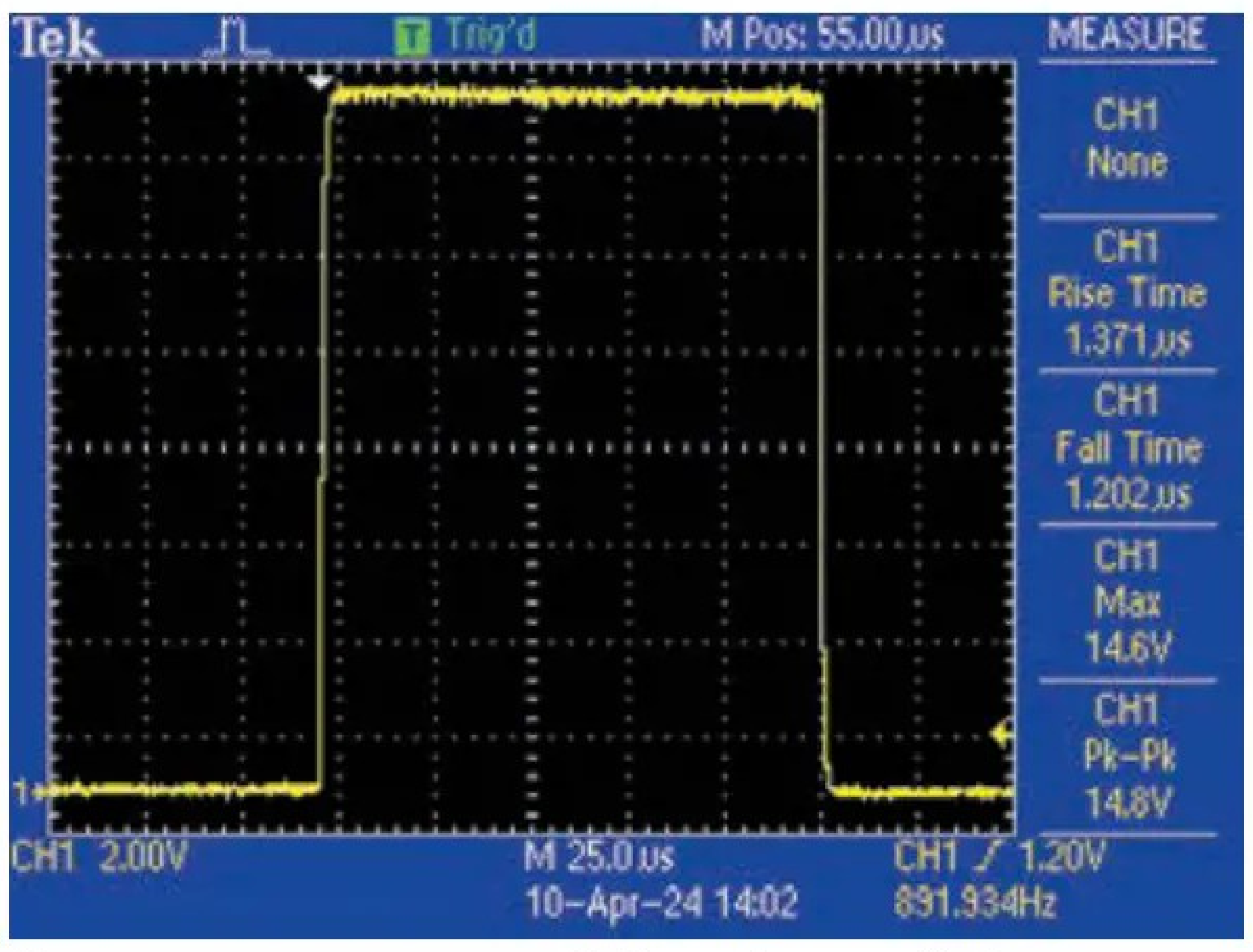
Speed trimpot VR7, in series with a 100kΩ resistor, provides level-shifting of the voltage from IC3c. As mentioned earlier, IC3c provides low-pass filtering with a corner frequency of around 16Hz due to the 1μF capacitor in parallel with the 100kΩ feedback resistor. IC3b provides voltage inversion with a gain of -2 before applying the speed feedback voltage to the N1 node via a 10kΩ resistor.

Motor speed adjustment

VR1 is used to adjust the motor



Scope 4 & 5: these captures show the sawtooth waveform and the Q1's PWM gate drive signal for duty cycles of about 10% and 90%. The top trace (yellow) is the PWM voltage, while the lower cyan trace is the sawtooth waveform with a range of ±5.6V. The dotted yellow horizontal line represents V_O; when the sawtooth waveform is below it, the PWM output is high.



Scope 6: a zoomed-in view of the PWM drive waveform at Q1's gate. The rise and fall times are 1.37μs and 1.2μs, respectively, with the gate voltage ranging from 0V when the IGBT is off to 14.8V when on.

speed manually. It is connected in series with a 620Ω resistor across the 12V supply. The 620Ω resistor is included so that the voltage from the potentiometer's wiper ranges from 0V to 10.6V. That provides a suitable range to feed to op amp IC2a, which is not a rail-to-rail type.

The potentiometer wiper charges and discharges a 100µF capacitor via one of two separate paths. When rotated clockwise to increase the voltage, the capacitor is charged via the 10kΩ resistor, so it takes about one second for the capacitor to fully charge and signal full motor speed.

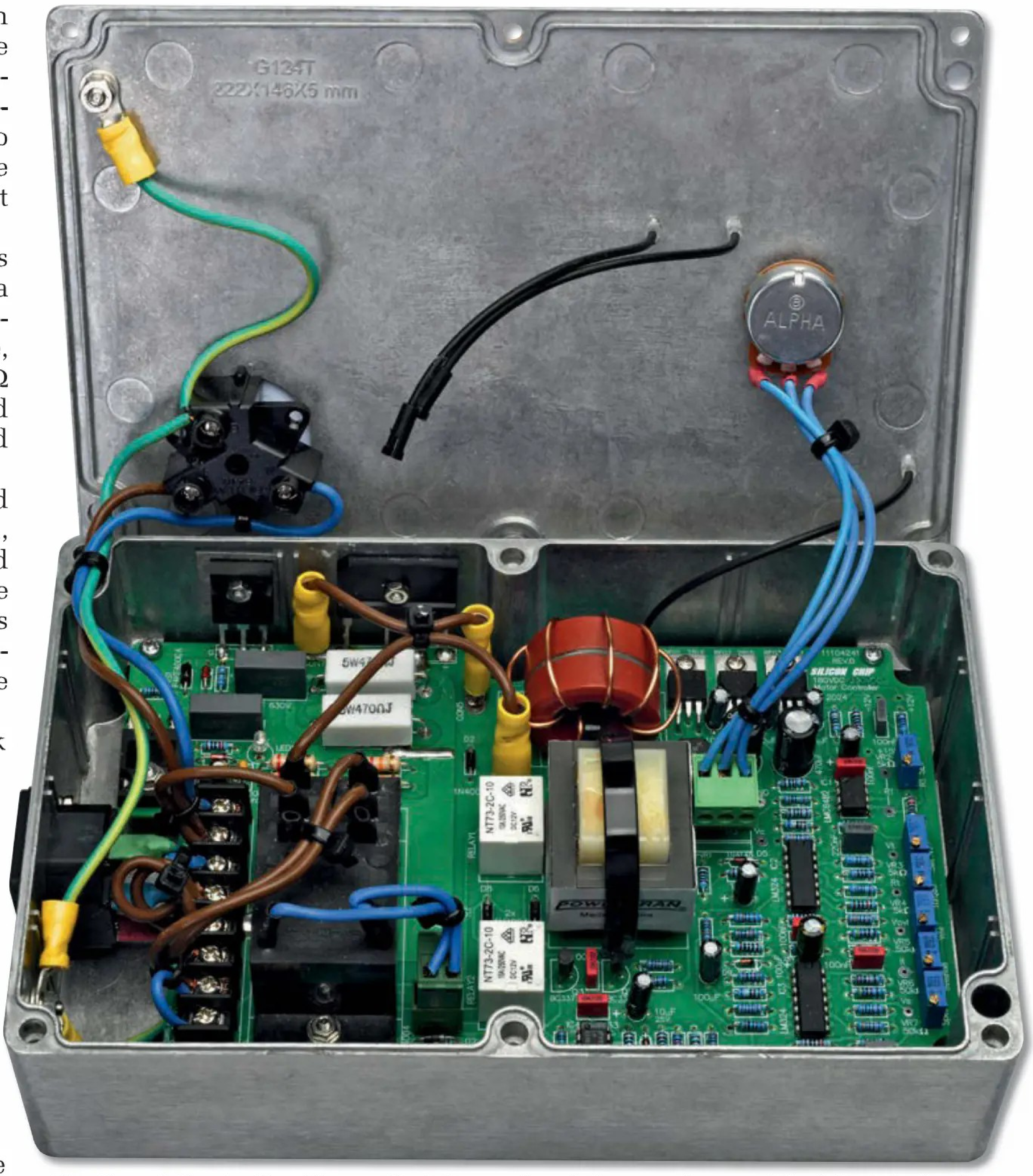
When the potentiometer is wound anti-clockwise to reduce the speed, the voltage is more quickly decreased by discharging the capacitor via diode D5 and its series 100Ω resistor. This allows the motor to be stopped quickly if necessary. IC2a's output feeds the N1 node via a 6.8kΩ resistor.

The current and voltage feedback signals, plus the speed control potentiometer signals, are all summed at the N1 node. This results in a summed output from the IC2b mixer of the three sets of voltages for the current and voltage feedback signals, plus the speed control potentiometer.

A second node (N2) mixes the IC2b output (via a 10kΩ resistor) and the torque from the IC2d output (via trimpot VR4 and its series 10kΩ resistor). The final summation is performed by IC2c, producing the V_O output signal that's applied to the PWM comparator, IC4a.

The IC4a comparator has a small amount of hysteresis so it does not oscillate when the non-inverting input voltage is close to the sawtooth oscillator waveform voltage. The 100Ω and 1MΩ resistors at pin 3 cause the pin 3 voltage to shift slightly when IC4a's output state changes, preventing the two voltages at pins 2 and 3 from remaining at the same level for long.

A voltage clamp comprising zener diode ZD3 and diode D10 at the pin 3 input to IC4a limits the voltage to one diode drop below -8.2V. This clamping prevents the input from going below the op amp's input voltage range. Without the clamp, if the voltage went below -9V, the op amp output would swing high instead of staying low due to a phase reversal internal to the op amp.



The Speed Controller fits neatly into an aluminium enclosure. The black 'wires' are actually fibre-optic light pipes for the LEDs.

The full adjustment range of VR1 suits 230V DC motors. For motors rated to a lower voltage, like 180V, you can simply operate VR1 over the lower 80% of its range. If you need to prevent more than an average of 180V from being applied to the motor, you can increase the 620Ω resistor in series with VR1 to 1.6kΩ.

Current overload detection

The motor-current-derived voltage from IC1b is applied to the current overload circuitry comprising op amp IC3d and comparator IC3a (another op amp used as a comparator). IC3d amplifies the voltage from IC1b with a gain of -4.68 times (220kΩ/47kΩ), with its output voltage level-shifted by trimpot VR5 that's connected to the -12V supply by way of a 24kΩ resistor.

IC3a compares the output voltage from IC3d against a reference voltage set by VR6.

VR6 is connected as an adjustable divider across the ±12V rails with a 12kΩ padder resistor and sets the current overload trip level.

If the motor current is high enough to produce a voltage from IC3d's output above the overload level, the comparator output will go high, switching on transistor Q3 and consequently, relay RLY2.

A 100µF capacitor holds this pin 2 input at -12V for a few seconds at power-up. The initial low voltage ensures the comparator output at pin 1 of IC3a is high at power up, so RLY2 switches on and opens its NC contact. That ensures RLY3 is not on during power-up, so the motor cannot run immediately.

When transistor Q3 is switched on, we ensure it's on long enough to activate RLY2 due to the 100µF capacitor that's initially charged via diode D9 from IC3a's output.

When there is an overcurrent condition and RLY2 is powered, it disconnects power to RLY3's coil. RLY3 is the high-current relay that connects power to the motor by joining the M+ terminal to inductor L1.

If an overcurrent condition triggers the relays, that will quickly cease as the motor will no longer be powered. RLY2's contacts will close again, so RLY3 can be powered once more, and the motor can be restarted.

However, power for RLY3's coil comes via RLY1's contacts, and RLY1's contacts are open unless the speed potentiometer is fully anti-clockwise. So, the speed potentiometer must be returned to the fully-off position before RLY1's contacts close. RLY3 is then powered to provide voltage to the motor once the speed potentiometer is rotated clockwise.

Restart switch

We described earlier how IC1a and RLY1 provide the 'switched potentiometer' action we need from a regular potentiometer, but here are more details on how that section works.

IC1a acts as a Schmitt-trigger comparator, monitoring the speed potentiometer voltage after buffer IC2a. It compares that voltage to a 119mV reference from a 100kΩ/1kΩ voltage divider across the 12V supply. When IC2a's output is below this 119mV reference, the output of IC1a is low, so RLY1 is not powered. When IC2a's output is above 119mV, IC1a's output goes high and drives transistor Q2 via its 1kΩ base resistor.

IC1a includes hysteresis so the output does not oscillate at the 119mV threshold. IC1a is powered from the 15V supply, with a 1MΩ feedback resistor, so this hysteresis is around 15mV.

In more detail, when IC1a's output is low, its pin 3 input is pulled lower than IC2a's output due to the 1MΩ/1kΩ voltage divider. When IC1a's output goes, pin 3 is pulled about 15mV higher, so the output from IC2a needs to drop a further 15mV before IC1a's output will go low again.

When Q2 is on and RLY1 is powered, its normally closed contacts open, disconnecting RLY3's 12V coil power

Parts List – 180-230V DC Motor Speed Controller

1 double-sided plated-through PCB coded 11104241, 201 × 134mm
1 Gainta G124 or BS11MF diecast aluminium enclosure, 222 × 146 × 55mm **[both available from TME]**
2 SRM-1C-SL-12VDC, NT73 -2 or equivalent 10A, 12V DC coil SPDT PCB-mounting relays (RLY1, RLY2) **[Farnell 1094019]**
1 T92P11D22-24, HF92F-024D-2C21S or FRA8PC-S2 30A, 24V DC coil DPDT panel-mount relay (RLY3) **[Farnell 270350]**
1 Myrra 44237 PCB-mounting 12V + 12V 5VA mains transformer (T1) **[Farnell 1214601]**
2 30 × 20 × 12.5mm powdered iron toroidal cores (L1) **[Farnell 4167313]**
1 8-way 300V 15A 8.25mm-pitch PCB-mount barrier terminals (CON1) **[Farnell 4476943]**
1 vertical-mount 300V 15A 3-way pluggable header with screw terminals, 5.08mm spacing (CON2) **[Farnell 2452499 + 3797901]**
1 vertical-mount 300V 15A 2-way pluggable header with screw terminals, 5.08mm spacing (CON3) **[Farnell 2468397 + 3399493]**
3 PCB-mount 5mm pitch 6.3mm male spade connectors (CON5-CON7)
3 6.3mm fully-insulated female crimp spade connectors
1 IEC C14 panel-mount mains connector with integral fuse (CON10) **[Farnell 9521615]**
1 M205 230VAC fast-blow fuse (F1) (with a current rating to suit the motor)
1 13A mains IEC lead
1 13A chassis-mount mains socket **[eg, RS 500-0459]**
1 24mm 5kΩ single-gang linear potentiometer, 500V rating (VR1) **[Farnell 1436327]**
3 5kΩ top adjust trimpots (VR2-VR4)
3 50kΩ top adjust trimpots (VR5-VR7
1 knob to suit VR1
2 14-pin DIL IC sockets (**optional**)
2 8-pin DIL IC sockets (**optional**)
3 100mm-long 3mm LED fibre-optic light transporters (**optional**) **[Mouser 749-FLP25V4.0-SBC]**
4 yellow 5mm inner diameter crimp eyelets for 4-6mm diameter wire
4 M3 × 6mm tapped standoffs **(not required for G124 case)**
8 M3 × 6mm panhead machine screws **(not required for G124 case)**

Hardware & cables
2 M4 × 10 panhead machine screws (for Earth-to-chassis connections)
2 4mm inner diameter star washers
2 M4 hex nuts
2 M3.5 × 6mm panhead machine screws (PCB to Altronics enclosure)
1 M3 × 12mm panhead machine screw (for Q1)
4 M3 × 10mm panhead machine screws (for BR1, D1 & RLY3)
2 M3 × 10mm countersunk head machine screws (for IEC connector)
3 M3 × 6mm panhead machine screws (for REG1-REG3)
3 3mm inner diameter washers (for D1 and RLY3)
8 M3 hex nuts
1 TOP3 insulating washer
1 500mm length of 1.25mm diameter enamelled copper wire
1 500mm length of 10A green/yellow striped (for Earth) mains-rated wire
1 500mm length of 10A brown (for Active) mains-rated wire
1 500mm length of 10A blue (for Neutral) mains-rated wire
1 450mm x 8mm plastic cable tie (for T1)
1 250mm x 4.8mm plastic cable tie (for L1)
12 100mm x 3.6mm plastic cable ties
1 120mm length of black 5mm diameter heatshrink tubing
1 60mm length of red 5mm diameter heatshrink tubing
1 15mm length of blue 5mm diameter heatshrink tubing
1 15mm length of yellow 5mm diameter heatshrink tubing

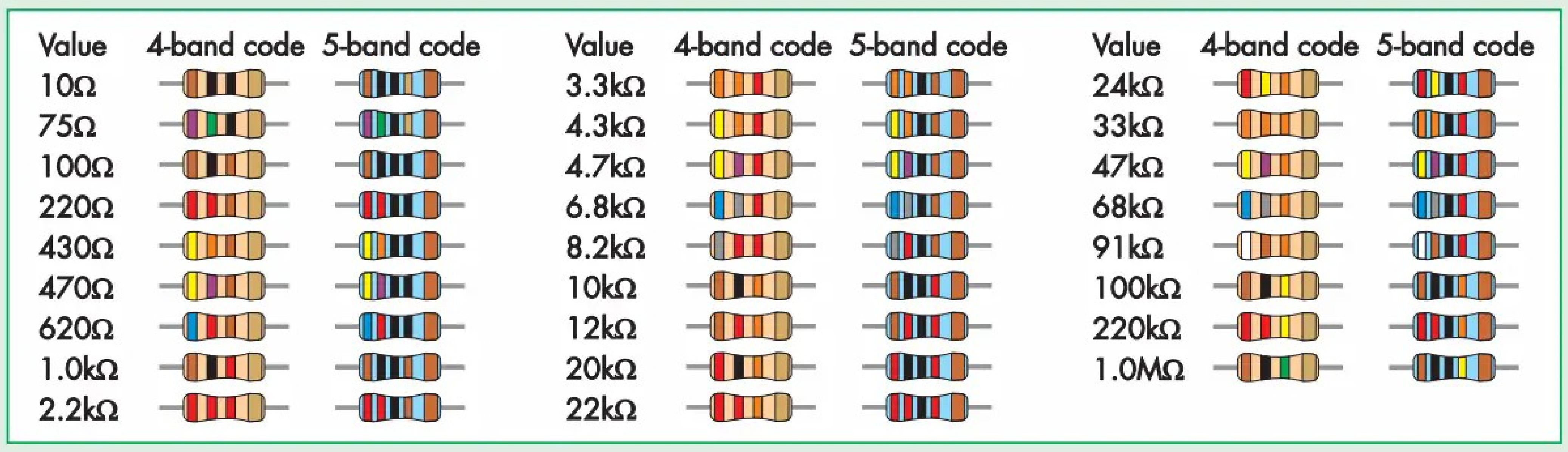
180-230V DC Motor Speed Controller

Semiconductors
1 LMC6482AIN dual CMOS op amp, DIP-8 (IC1) **[Farnell 3117147]**
2 LM324AN quad op amps, DIP-14 (IC2, IC3)
1 LM833 dual op amp, DIP-8 (IC4)
1 TLP5701 optically-isolated Mosfet driver, SMD-6 (IC5) **[Farnell 3872508 or 2768341]**
1 7812 +12V 1A linear regulator, TO-220 (REG1)
1 7815 +15V 1A linear regulator, TO-220 (REG2)
1 7912 -12V 1A linear regulator, TO-220 (REG3)
1 STGW40M-120DF3 1.2kV 80A IGBT, TO-247 (Q1) **[Farnell 2470028]**
2 BC337 NPN transistors, TO-92 (Q2, Q3)
4 3mm or 5mm high brightness red LEDs (LED1, LED3-LED5) ◆
1 3mm or 5mm high-brightness green LED (LED2) ◆
1 PB5006 600V 45A SIL bridge rectifier (BR1) **[Farnell 3774973]**
1 W04 1A 400V bridge rectifier (BR2)
1 RURG3060 600V 30A fast diode (D1) **[Farnell 2495903]**
4 1N4004 400V 1A diodes (D2, D6-D8)
5 1N4148 75V 200mA signal diodes (D3-D5, D9, D10)
2 15V 1W zener diodes (ZD1, ZD2) **[1N4744A]**
1 8.2V 1W zener diode (ZD3) **[1N4738A]**
1 P4KE400CA bi-directional TVS diode (TVS1) **[Mouser 576-P4KE400CA]**
◆ *3mm diameter required if light transporters are used*

Capacitors
2 470µF 25V PC electrolytic
4 100µF 16V PC electrolytic
1 100µF 25V PC electrolytic
4 10µF 25V PC electrolytic
1 1µF 63V or 100V MKT polyester
1 1µF 50V multi-layer or monolithic ceramic
1 220nF 63V or 100V MKT polyester
7 100nF 63V or 100V MKT polyester
1 100nF X7R multi-layer or monolithic ceramic
2 47nF 630V pulse double-metallised polypropylene (Kemet R76PI24705050J) **[Farnell 3649826]**
1 10nF 63V or 100V MKT polyester

Resistors *(all axial ¼W 1% unless noted)*
2 1MΩ
1 220kΩ
1 220kΩ 1W (5% OK)
8 100kΩ
1 91kΩ
1 68kΩ
2 47kΩ
3 33kΩ
1 24kΩ
4 22kΩ 1W (5% OK)
2 20kΩ
1 12kΩ
8 10kΩ
1 10kΩ (SMD 1206-size)
2 8.2kΩ
2 6.8kΩ
4 4.7kΩ
1 4.3kΩ
1 3.3kΩ
2 2.2kΩ
4 1kΩ
2 620Ω
2 470Ω 5W (5% OK)
1 430Ω
1 220Ω
4 100Ω
1 75Ω
1 10Ω
2-4 0.022Ω 3W 1% SMD M6332/2512-size (TE Connectivity TLRP3A30CR022FTE) **[Farnell 3828731]** ●
0-2 0.05Ω 3W 1% SMD M6332/2512-size (TE Connectivity TLRP3A30CR050FTE) **[Farnell 3828740]** ●

● *see Table 1 next month for quantities (4 × 0.22Ω is sufficient for all motors)*



Value
4-band code
5-band code
10Ω
75Ω
100Ω
220Ω
430Ω
470Ω
620Ω
1.0kΩ
2.2kΩ
3.3kΩ
4.3kΩ
4.7kΩ
6.8kΩ
8.2kΩ
10kΩ
12kΩ
20kΩ
22kΩ
24kΩ
33kΩ
47kΩ
68kΩ
91kΩ
100kΩ
220kΩ
1.0MΩ

unless RLY3 is already on and its contacts are closed. This is because a set of RLY3's contacts are in parallel with RLY1's contacts (the points labelled 'a' and 'b' in Fig.2).

The only way to restore power to the motor via the RLY3 contacts is to return speed potentiometer VR1 to its fully anti-clockwise position. In this case, RLY1's contacts close and +12V is reconnected to RLY3's coil.

The safety switch connection between pins 7 and 8 of CON1 can also stop the motor and prevent it from restarting until the speed potentiometer is returned to the anti-clockwise position.

An open safety switch disconnects power to RLY3's coil, immediately removing power to the motor.

Setting it up safely

You might be wondering about the purpose of the CON5 & CON7 terminals near CON1 on the circuit diagram, shown joined by a wire bridge. This allows you to disconnect the +12V supply from the positive terminal of the bridge rectifier when making adjustments.

Also, because the mains supply to the active side of the bridge rectifier (BR1) and transformer T1 are via separate terminals on CON1, BR1 can be left disconnected during initial setup and testing.

With BR1 disconnected, the motor can't run, and much of the circuit is essentially isolated from the mains Active. This allows you to adjust some of the trimpots and monitor the voltages in the circuit more safely. The circuit is still connected to mains Neutral via the bridge rectifier, though. So, during setup, it is essential to check that the mains Neutral is close to the Earth voltage.

Even though some adjustments can be made with the mains Active isolated, some trimpots must be adjusted while the circuit is at mains potential. We will describe how to do this safely in the setup and testing section of the article next month. It involves using a high-voltage-insulated screwdriver with a multimeter and probes rated for use at mains voltages.

Next month

Construction, testing and setup details for the 180-230V DC Motor Speed Controller will be in a follow-up article next month.

By Andrew Levido

Precision Electronics

Part 7: Analog-to-Digital Conversion

Last month, in the sixth instalment in this series, we covered the various sources of analog-to-digital and digital-to-analog conversion errors. We also looked at digital-to-analog converters (DACs) in detail. This month, we will focus on analog-to-digital converters (ADCs) and, as usual, that will include a practical example.

Just as a quick recap, we saw that all converters exhibit quantisation errors due to the discrete way numbers are represented in digital systems. Quantisation error is directly related to the number of bits (the resolution) of the converter. We saw that this can cause quantisation noise if we are dealing with AC signals.

On top of quantisation errors, we saw that there are usually offset, gain and non-linearity errors associated with conversion and that these can be combined to give a total unadjusted error (TUE) figure that can be used in error calculations. All of this applies equally to DACs and ADCs.

Sampling and aliasing

In contrast to DACs, which convert discrete digital codes to discrete voltage levels, ADCs have to convert an infinitely variable (and maybe varying) voltage level to discrete digital codes. We therefore have to 'sample' the analog voltage at some instant in time and convert that value to the appropriate (nearest) digital code.

Because the conversion takes a finite amount of time, in most instances, we want to take a 'snapshot' of the input voltage so that the entire conversion process takes place with a fixed input value. For this reason, many converters (but not all, as we shall see below) are preceded by a 'sample-and-hold' circuit similar to that shown in Fig.1.

The output of the sample-and-hold buffer follows the input during the sampling period, when the switch is closed. It is held constant by the capacitor during the hold period, while the switch is open. The conversion takes place during the hold period while the value is stable.

In Fig.1, I have shown the sample and hold's output (red trace) instantaneously snapping back to track the input voltage when the switch is closed. In reality, the capacitor takes a finite time to charge or discharge.

If your ADC has a sample-and-hold system, you need to make sure the sampling time is long enough for the capacitor to fully charge to the signal voltage through the signal's source impedance. This source impedance can include the sample-and-hold switch on-resistance, the on-resistance of any analog multiplexer, and the external source impedance. Values in the kilohms range are not unusual.

The sampling time should be long enough for the capacitor voltage to charge or discharge to within ½LSB (least significant bit) of the signal voltage to avoid adding error to the conversion. Many ADCs allow the user to control the sampling time for this purpose.

Assuming we want to perform the analog-to-digital conversion on an ongoing basis, we need to sample and convert the input signal at regular intervals. We call these intervals the sampling rate.

The Nyquist-Shannon sampling theorem states that an AC signal can be fully reconstructed (without any loss whatsoever) so long as the sampling rate, f_{SAMP}, is at least twice the highest frequency component present in the signal (f_{MAX}). The particular sampling rate that is exactly twice f_{MAX} is the known as the Nyquist frequency, f_N.

If we sample at a higher rate than strictly necessary ($f_{SAMP} > f_N$), we are said to be oversampling, while if we sample at a lower rate ($f_{SAMP} < f_N$), we are undersampling. We often want or need to oversample, but we generally try to avoid undersampling as it can lead to a phenomenon called aliasing, which can give rise to significant errors.

Fig.2 shows what can happen if we

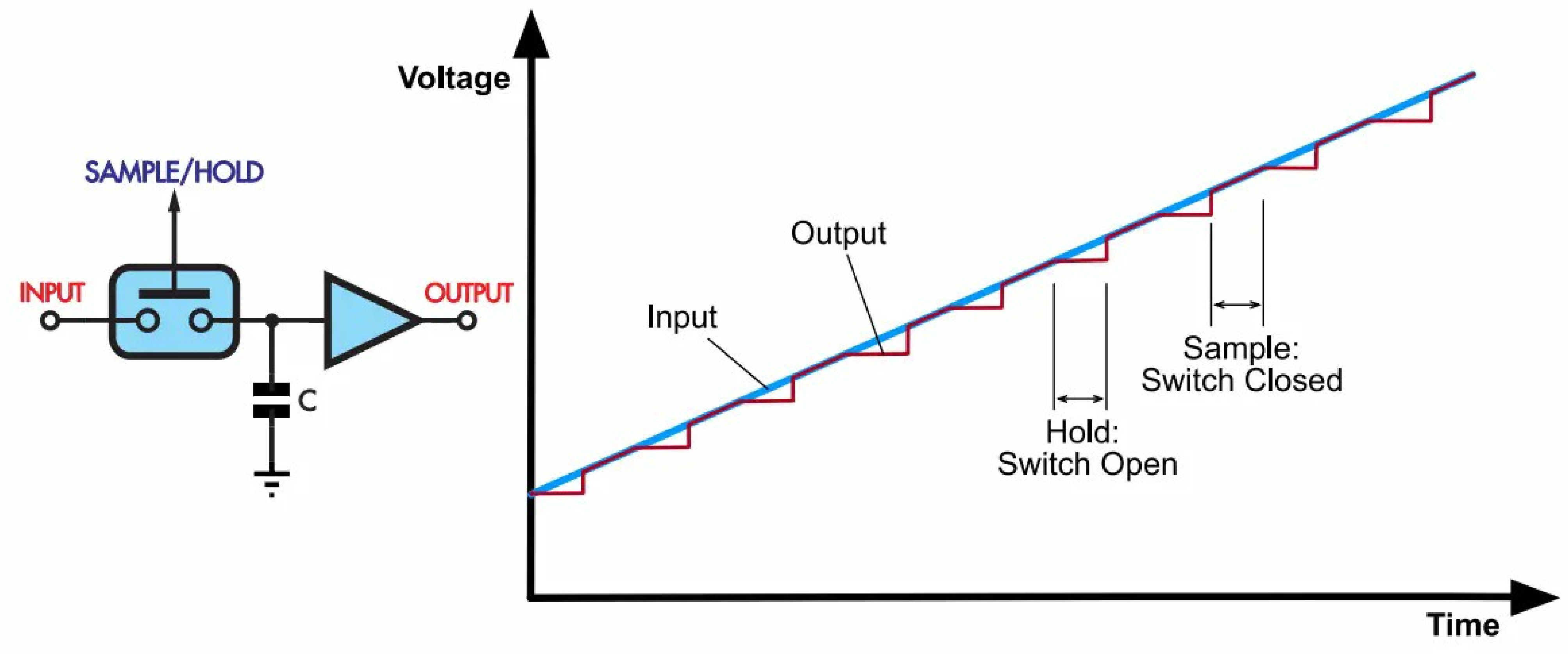


Fig.1: the output of a sample-and-hold circuit follows the input while the switch is closed, but 'freezes' the value while it is open. This allows the analog-to-digital conversion process to occur with a steady input voltage.

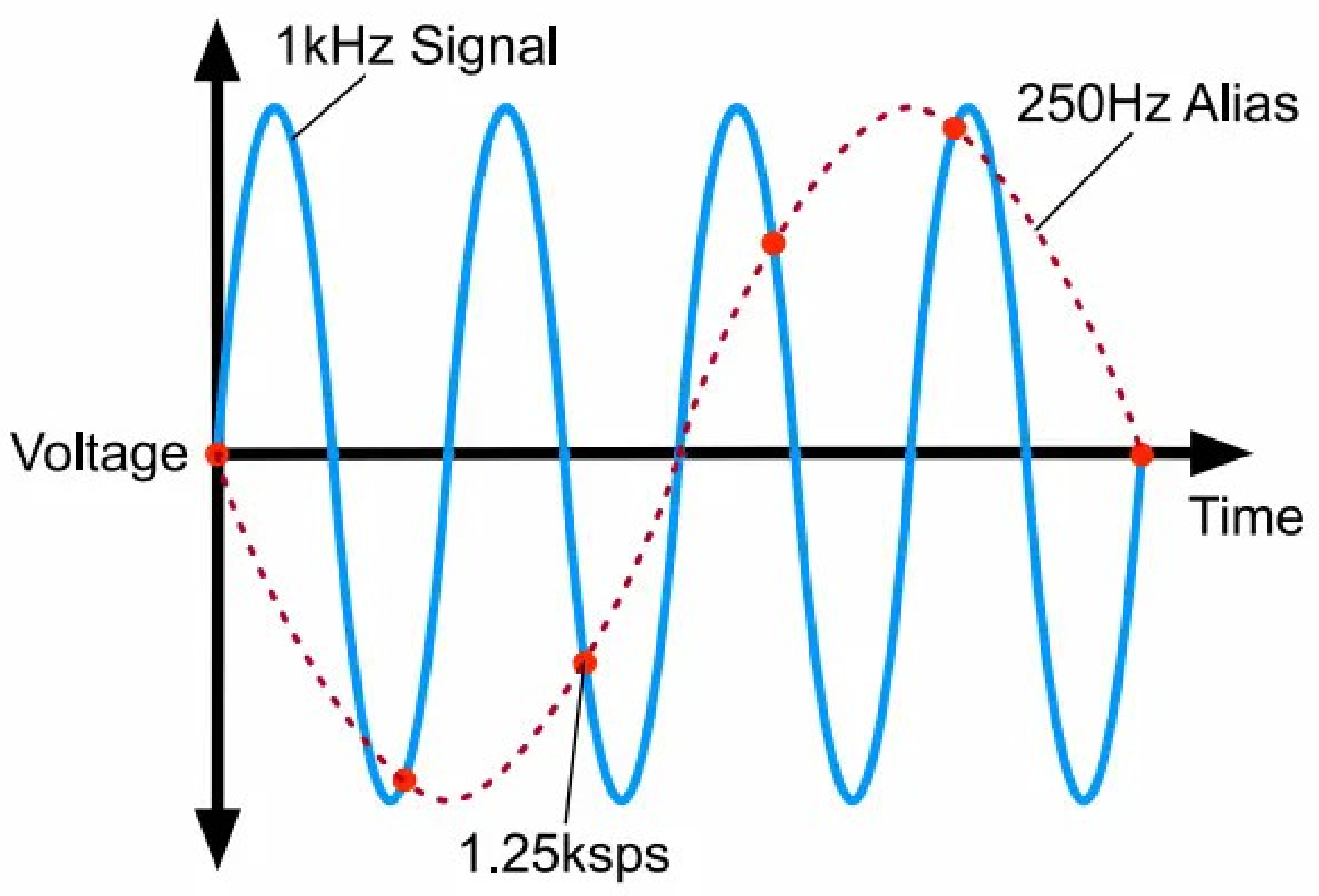


Fig.2: here a 1kHz signal is sampled at 1.25ksps, lower than the Nyquist limit of 2ksps. This results in the ADC measuring a 250Hz alias signal instead of the expected 1kHz signal.

undersample a signal. Here, a 1kHz sinewave (shown in blue) is sampled at about 1.25ksps – lower than the Nyquist frequency of 2ksps. The sample points (red dots) trace out a false 'alias' signal with a frequency of 250Hz.

Within the digital system, we will have no idea that the true signal includes a 1kHz component and that the 250Hz signal is an alias – all we will measure is the 250Hz sinewave.

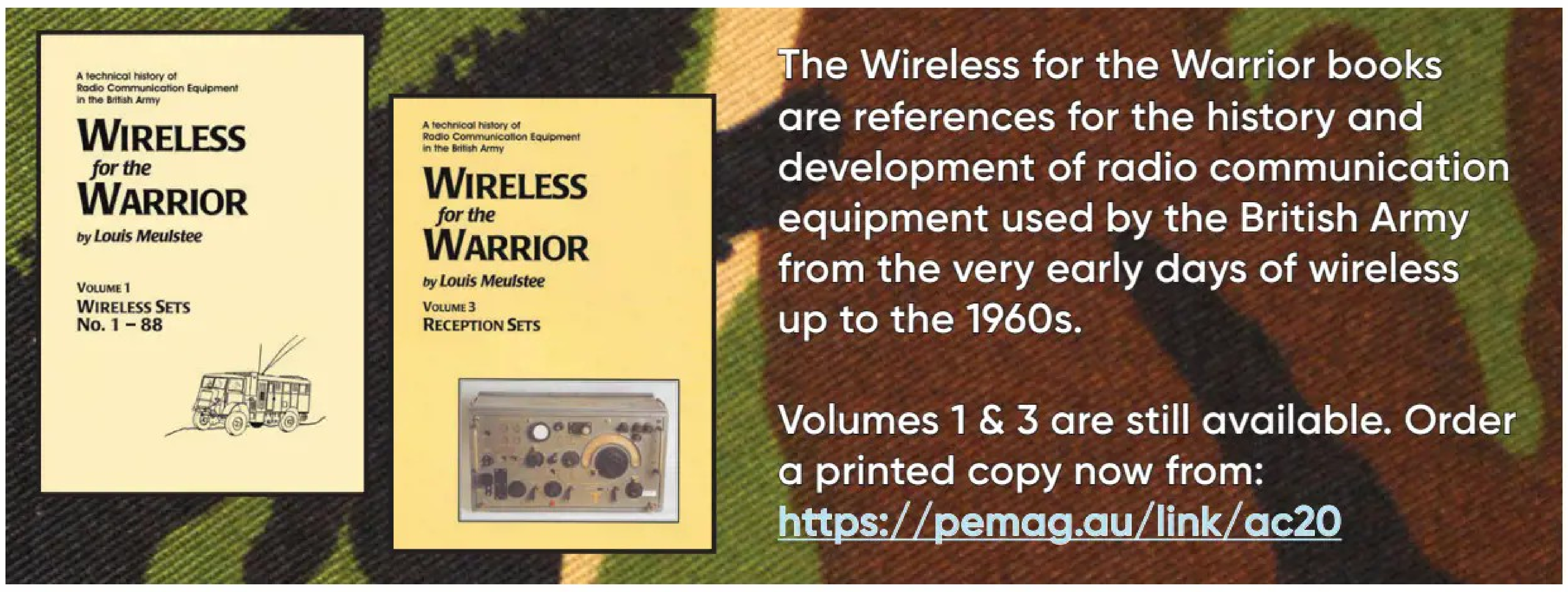
To avoid aliasing, we must ensure that there is no content in the sampled signal with a frequency higher than ½f_{SAMP}. This can be achieved by limiting the bandwidth of our signal with a filter, or by using a high enough sampling rate.

In practice, we often need to do both. For wideband signals, it can be difficult or impossible to totally eliminate aliasing since perfect 'brick wall' filters are hard to come by! Instead, we have to be satisfied with reducing the amplitude of the worst possible alias to something we can live with.

It is easiest to understand this in the frequency domain, as shown in Fig.3. In each diagram, the vertical axis is the relative amplitude of the signal in decibels (dB), while the horizontal axis is the frequency on a linear scale. We are interested in digitising a broadband signal within the band of interest shown shaded in blue.

Because the signal is broadband, we apply some low-pass filtering with corner frequency f_C, shown by the solid curve. The dotted lines represent the magnitude of the alias signals obtained by reflecting the filter roll off about the Nyquist limit (½f_{SAMP}). If the filter had a 'brick wall' cutoff, there would be no aliasing.

In the first diagram, we try to eliminate aliasing by adding a second-order low-pass filter, with its corner frequency set at the upper end of the bandwidth of interest, and by oversampling by 50% (the Nyquist limit is set 50% higher than the upper limit of the band of interest). You might think that this would be enough to eliminate aliasing, but unfortunately, it is not.


The Wireless for the Warrior books are references for the history and development of radio communication equipment used by the British Army from the very early days of wireless up to the 1960s.

Volumes 1 & 3 are still available. Order a printed copy now from: https://pemag.au/link/ac20

The dotted line shows that there will still be an alias component within the band of interest, although it will be 6dB or more below the level of the signal of interest. This happens because there is still content in the low-pass filtered signal with frequency components above the Nyquist limit, albeit at a low level.

The second chart shows that using a fourth-order filter, with its steeper roll-off, helps by shifting the alias signal down to -16dB or lower. We could improve this even further by using a sixth- or eighth-order filter, at the expense of complexity.

The final chart shows what happens if we retain the fourth-order filter but increase the sampling rate to oversample at 100%, rather than the 50% in the first two cases. The alias signal is now down by 30dB or more.

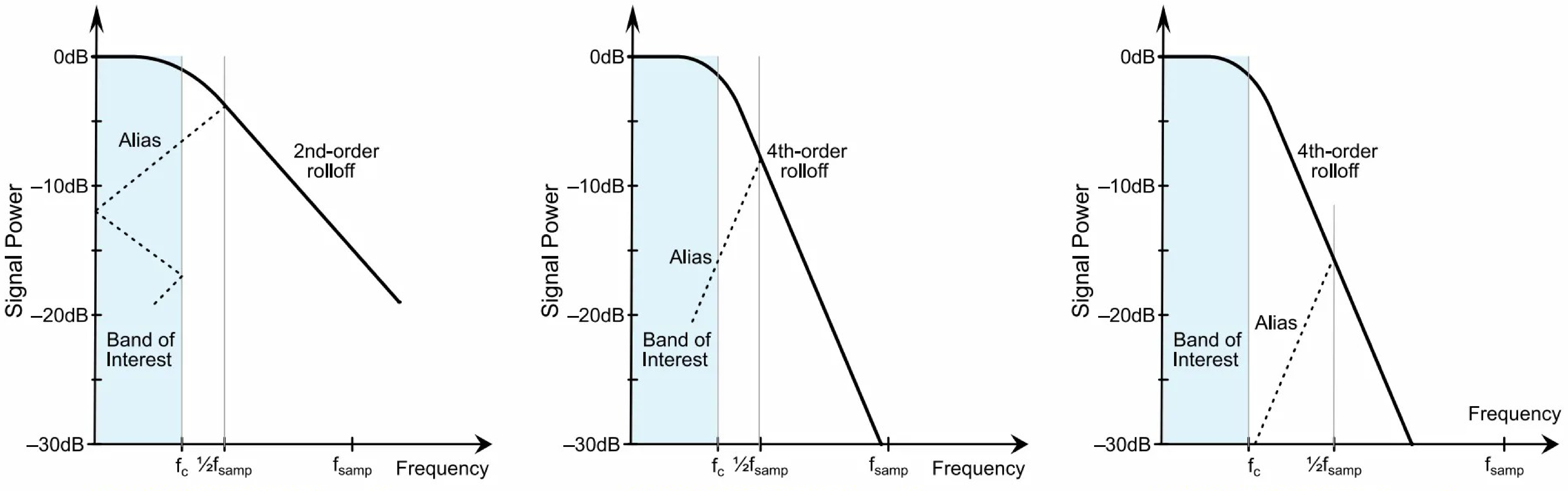
The long and short of this is that if you are digitising signals with a broadband AC component, you need to choose your sampling rate and filter configuration carefully to ensure you don't introduce errors due to aliasing. You don't have to eliminate the aliases entirely – you just need to get them down to a level where the errors from them are manageable.

You may not need an anti-aliasing filter or oversampling if your signal is band-limited by its very nature. If your signal is nominally DC (or at least very slowly changing), you can be even more relaxed about choosing the sampling rating and anti-aliasing filter.

Flash ADCs

With anti-aliasing taken care of, and a sample-and-hold system keeping our ADC's input constant while sampling, we are ready to actually convert our analog signal to a digital one. The most straightforward way to do this is the 'flash' or parallel ADC, a simple three-bit example of which is shown overleaf in Fig.4 (this is not directly related to flash memory).

A resistor string establishes a series of threshold voltages representing the transition voltages between each code. The input voltage is simultaneously compared to all of these thresholds. A comparator output will be asserted low if the input voltage exceeds the respective transition threshold. A priority encoder outputs the code associated with the highest-value input that is asserted.



(a) SECOND-ORDER FILTER, $f_{samp} = 3f_c$

(b) FOURTH-ORDER FILTER, $f_{samp} = 3f_c$

(c) FOURTH-ORDER FILTER, $f_{samp} = 4f_c$

Fig.3: using a higher-order low-pass filter and increasing the sampling rate can both help reduce aliasing when digitising broadband signals.

True flash converters require 2^n matched resistors and the same number of comparators (where n is the resolution in bits), so they are usually limited to about 16 bits, but they are very fast.

They can make a conversion every clock cycle, so they can reach sampling rates in the Gsps (gigasamples per second or 1,000,000,000+ samples per second) range.

Many modern flash ADCs use a multistage architecture with a series of lower bit-count flash conversions of increasing precision. These 'pipelined' flash converters can have a latency of several tens of clock cycles, but maintain conversion rates in the Gsps range, since sequential samples are being processed in each stage.

Several flash ADCs can be interleaved to achieve even higher sampling rates. If you have a digital oscilloscope (DSO), it most likely uses a pipelined flash converter with eight, 10 or 12 bits of resolution and a multi-Gsps sampling rate.

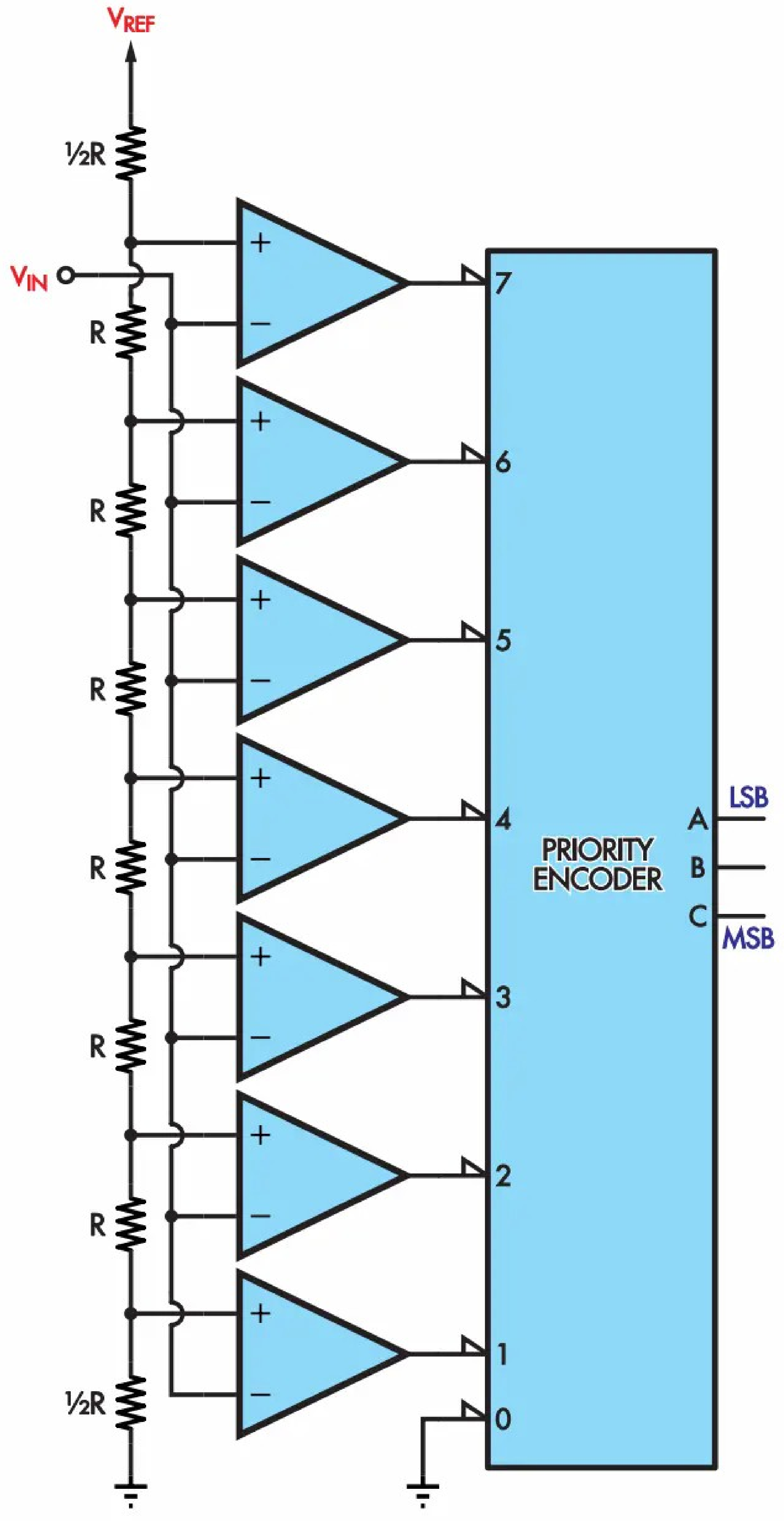


Fig.4: a 'flash' ADC compares the input signal to each threshold voltage simultaneously. The output is the digital code associated with the highest threshold the input signal reaches.

Successive approximation

Successive approximation analog-to-digital conversion uses a binary search strategy to find the digital code corresponding to the analog input. A simplified three-bit successive approximation converter is illustrated in Fig.5.

At the start of the conversion cycle, the controller clears the successive approximation register and sets its MSB to one. The output of the DAC will therefore be a voltage that is 50% of the full scale. The comparator checks if the input voltage is above or below this threshold. If it is above 50% (comparator output high), the MSB in the SAR remains set; otherwise, it is cleared.

The controller then sets the next most significant bit so the DAC and comparator can check if the input voltage lies in the upper or lower part of the appropriate sub-range. Again, the bit remains set or is cleared based on the comparator output. This process continues bit-by-bit until the value of the least significant bit is confirmed.

At this point, the controller latches the SAR contents through to the converter output, and the cycle can begin again.

This binary search process is shown graphically on the right side of Fig.5. Starting with the MSB, each bit is set or cleared successively until the output code is complete. This iterative approach means that the conversion takes at least one clock cycle for each bit, so SAR converters are generally slower than flash converters, with conversion rates typically limited to the Msps (megasamples) range.

This is the type of converter you will usually find in microcontrollers and many low-cost serial interface ADC chips.

Integrating converters

You can see why a sample-and-hold system is important if you are using a successive approximation ADC. If the value of the input were to change mid-conversion, the result could be a wrongly set bit and therefore a potentially significant error.

However, there is a class of converter – the integrating converter – that can accommodate a changing input during the conversion cycle. In fact, we can use this characteristic to our advantage.

The simplest integrating converter is the single-slope variant shown in Fig.6. When the start signal is pulsed, the flip-flop is set, the transistor is switched off and capacitor C begins to charge linearly at a rate determined by the value of the current source. Simultaneously, the clock is gated through to the counter, which begins counting up.

When the voltage on the capacitor rises to V_{IN}, the comparator resets the flip-flop and stops the counter, which holds a number proportional to the time taken to charge the capacitor to V_{IN}. The count time is given by $T_{COUNT} = C \times V_{IN} \div I$ and the count value will be $N = (C \times V_{IN}) \div (I \times T_{CLK})$, where T_{CLK} is the clock period.

The output count is therefore dependent not only on V_{IN} and the current I but also on the capacitor value and the precision of the clock frequency. The latter two are a bit of a problem, since tight tolerance capacitors are rare and expensive, and very precise and stable clocks are not easy to create.

Fortunately, a variation on this scheme – the dual-slope converter – solves these problems very elegantly. Fig.7 shows how it works. This time, the controller first charges the capacitor up for a fixed period (T_{CHARGE}) with a current proportional to the input voltage.

The capacitor is then discharged by a fixed current of I_{DIS}, and the time taken for the capacitor voltage to ramp down to zero (T_{COUNT}) is measured by the counter.

If we allow the capacitor to charge for M clock cycles, its voltage will reach $V_{CAP} = I_{IN} \times M \times T_{CLK} \div C$. The count required for the capacitor to discharge from V_{CAP} to 0V will be $N = C \times V_{CAP} \div (I_{DIS} \times T_{CLK})$.

As V_{CAP} is the same for both charge and discharge phases, we can substitute the first equation into the second. The capacitor value and the clock period cancel out, and we are left with $N = M (I_{IN} \div I_{DIS})$. Recalling that I_{IN} is proportional to V_{IN}, we can see the count N is proportional to V_{IN}, I_{DIS} and some constants.

The conversion precision is therefore not dependent on the capacitor's value or the clock frequency. As a bonus, any offset error in the comparator is also eliminated since the charge-discharge cycle will start and end at the same voltage, even if it is not precisely zero.

Part 7 - analog-to-digital conversion

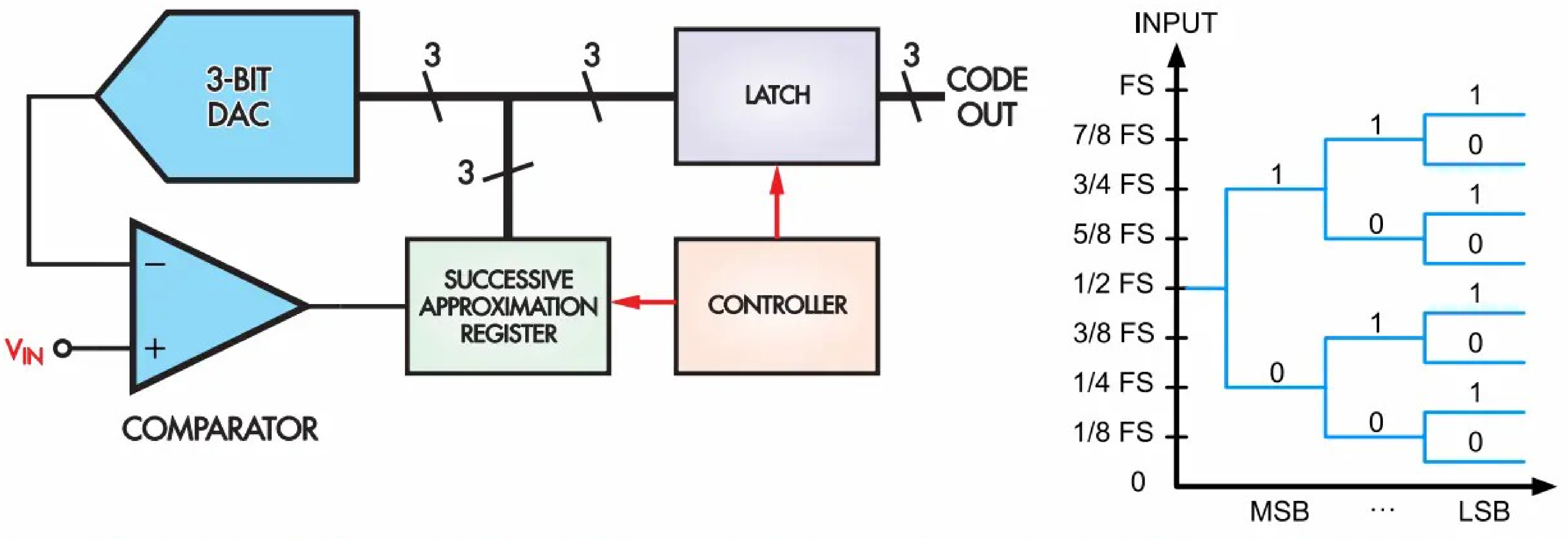

3-BIT DAC
3
3
3
LATCH
CODE OUT
SUCCESSIVE APPROXIMATION REGISTER
CONTROLLER
COMPARATOR
VIN
INPUT
FS
7/8 FS
3/4 FS
5/8 FS
1/2 FS
3/8 FS
1/4 FS
1/8 FS
0
1
0
MSB
···
LSB

Fig.5: a successive approximation ADC uses a binary search algorithm to determine the state of each successive bit, starting with the MSB (most significant bit).

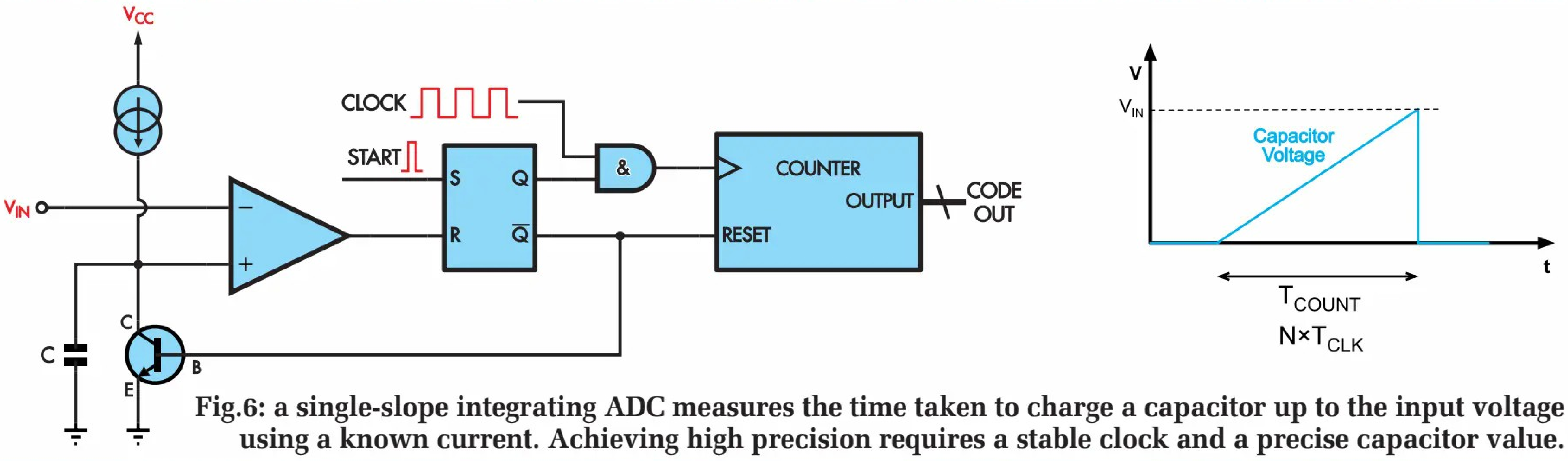

VCC
CLOCK
START
VIN
S
Q
R
Q̅
&
COUNTER
OUTPUT
RESET
CODE OUT
C
B
E
C
V
VIN
Capacitor Voltage
t
TCOUNT
N×TCLK

Fig.6: a single-slope integrating ADC measures the time taken to charge a capacitor up to the input voltage using a known current. Achieving high precision requires a stable clock and a precise capacitor value.

The only requirement is to use precision current sources and a capacitor with low dielectric absorption (a polypropylene dielectric is a good choice). Dielectric absorption is the mechanism responsible for the 'memory effect' in capacitors, where a recently discharged capacitor recovers some voltage over time after being discharged. This would obviously lead to errors in the dual-slope converter.

The 'integrating' nature of the charge cycle explains why dual-slope ADCs don't generally need a sample-and-hold circuit. Any changes in input voltage are averaged out over the capacitor charge period. This means integrating converters are inherently low-pass filters, so they work best with DC or very low-frequency signals. An anti-aliasing filter is not normally required for the same reason.

You can take advantage of this averaging to very effectively reject any mains-frequency interference that might be present on your signal. By setting the charge time to an integer multiple of the mains cycle period (20ms for 50Hz mains), any mains component present at the input will be averaged to zero over one or more full cycles.

Being counter-based, integrating ADCs are quite slow, but with resolutions of 20 or more bits (better than 1ppm resolution), and the ability to effectively reject mains interference, they are widely used in test and measurement equipment like digital multimeters (DMMs).

Very high-end test equipment (6½ or 7½ digit multimeters, for example) use more advanced variants known generically as multi-slope converters. Each manufacturer has their own proprietary flavour, but they all rely on the same charge-balancing principle.

Delta-Sigma ADCs

Another type of ADC that has come to the fore in recent years is the delta-sigma converter. A delta-sigma ADC consists of an analog modulator that produces a single bit stream, followed by a complicated digital filter. The inner workings of delta-sigma ADCs are not easy to describe or understand – so bear with me as I give it a shot.

We will start with the modulator, which is where the magic happens. The upper part of Fig.8 shows a simplified first-order modulator. Practical ADCs use higher-order modulators, but the principles remain the same.

The input is an analog voltage in the range ±V with a maximum frequency

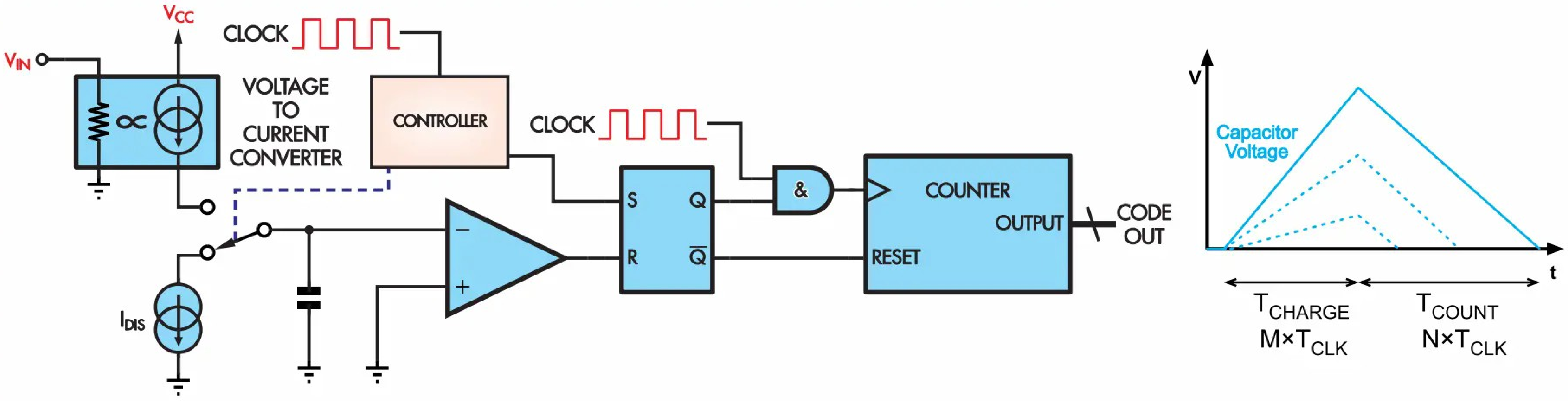

VIN
VCC
CLOCK
VOLTAGE TO CURRENT CONVERTER
∝
CONTROLLER
IDIS
CLOCK
S
Q
R
Q̅
&
COUNTER
OUTPUT
RESET
CODE OUT
V
Capacitor Voltage
t
TCHARGE
M×TCLK
TCOUNT
N×TCLK

Fig.7: the dual-slope integrating ADC has the advantages of not being dependent on either a precise capacitor value or clock frequency. Its integrating nature also allows it to be configured to reject mains interference.

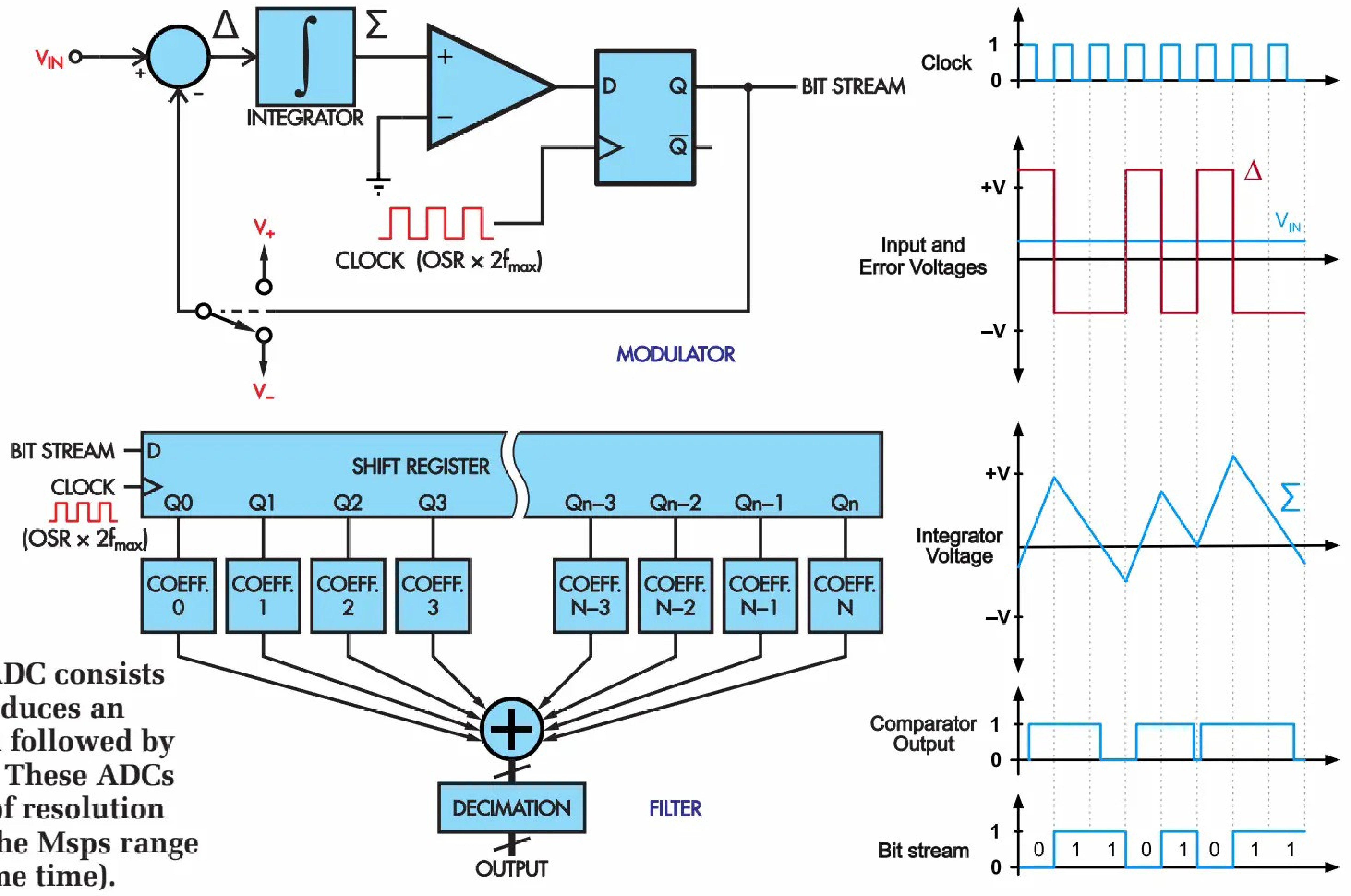


Fig.8: the delta-sigma ADC consists of a modulator that produces an oversampled bit stream followed by a complex digital filter. These ADCs can have up to 32 bits of resolution and sampling rates in the Msps range (although not at the same time).

component of f_{MAX}. The modulator is clocked at a rate higher than $2f_{MAX}$ by an oversampling rate factor (OSR). We will assume the OSR is 128 for the purposes of this example. The clock frequency is therefore $256f_{MAX}$.

The output of the modulator is a stream of 1s and 0s at the clock frequency, where a 1 code corresponds to V+ and a 0 code corresponds to V–. The average value of this bit stream over many cycles is equal to V_{IN}. For example, a zero-volt input would be represented as a string of alternating 1s and 0s corresponding to alternating V+ and V– voltages, averaging to 0V.

The example waveforms to the right of the figure show what happens with an input voltage of ¼V+. In the initial clock cycle, the bit stream value is zero, and the switch directs a voltage of V– to the summing junction, where it is subtracted from the input voltage.

The resulting voltage (5⁄4V) represents the 'error' between the input and the reconstructed modulator output (this is the 'delta' part of the delta-sigma converter). This error is integrated (the 'sigma' part) and the comparator determines if the result is positive or negative. In our example, the result transitions from negative to positive about ¼ of the way through the clock cycle.

On the next clock edge, a one is latched into the bit stream. The error voltage swings to -¾V, and the integrator starts to ramp down. The result is still positive at the end of this cycle, so the comparator output stays high and the third bit in the stream is also a 1. This process continues indefinitely, producing a bit stream with five 1s and three 0s for every eight bits, as shown.

Since we want to provide an output code at the sampling frequency ($2f_{MAX}$), we have to do it every 128 clock cycles. There can therefore only be 128 possible discrete values in the bit stream for each sample (128 zeros to 128 ones).

If this was all there is to it, we would have created a 7-bit converter, which is pretty unexciting. However, the delta-sigma converter has a trick or two up its sleeve.

The filter component of the ADC is a digital filter called a Finite Impulse Response (FIR) filter. We could write a whole series of articles on digital filters, but for a one-bit input this just consists of a long shift register with each output enabling or disabling a coefficient (a carefully chosen number) depending on whether it is a one or a zero.

All the coefficients are summed on each clock cycle to produce a digital output code. The coefficients are chosen to produce a very steep low-pass filter, with a cutoff frequency of f_{MAX}.

The output is decimated so that the resulting number changes only once for each sampling period. Decimation in our example just means each 128th sample is sent to the output and the other 127 are thrown away.

The filter coefficients have a resolution much higher than seven bits, and there may be many hundreds or even thousands of coefficients in the filter. This means each output code can take significantly more than 128 different values and, therefore, it has much more than seven bits of resolution.

If you find this last part hard to grasp, you are not alone. The mathematics behind it is complex, and some of the explanations you will find are confusing.

Another way to look at it is to think of the modulator as a 'perfect' ADC with significant quantisation noise superimposed on it. The oversampling nature of the modulator is such that this noise is 'shaped' (pushed up) to frequencies well above the sampling rate.

The low-pass filter then blocks most of this noise, leaving a level of quantisation noise corresponding to many more bits of resolution than the oversampling rate would suggest.

Delta-sigma ADCs offer excellent performance at reasonable prices. Audio ADCs can easily have 24-bit resolution and sampling rates of 96kbps or 192kbps, with extremely low distortion. Precision DC-accurate delta-sigma converters with up to 32 bits of resolution are available (at a price).

Delta-sigma ADCs are available with sampling rates up to 20Mbps. One of the big advantages of delta-sigma converters is their inherently high oversampling rate means that anti-aliasing filtering is made easier.

A practical example

I am developing a project that uses a low-cost microcontroller with a 12-bit successive approximation ADC to measure a ±6V analog signal. We will use this example to see what kind of performance we can expect from this pretty common scenario.

The microcontroller I have chosen is the STM32L031, a low-power, low-pin-count unit with a Cortex M0+ CPU core. It has a built-in 12-bit ADC with a maximum sample rate of 1.1Msps that uses the microcontroller's power rail as its reference.

The ADC's headline specifications are modest, with a worst-case offset error of ±2.5LSB, a worst-case gain error of ±2.0LSB and an INL of ±2.5LSB for a TUE of just over ±4LSB. Data sheet typical values are about half of these figures, but you already know how I feel about typical values.

This would mean that the lower two bits of the result probably should not be trusted, making this effectively a 10-bit converter, unless we can do something to improve its performance.

The ADC does have some nice features. One of the downsides of microcontroller ADCs is that they use the power rail (or, if you are lucky, a dedicated analog supply pin) as the full-scale voltage reference. Since I am powering this device from a 3V coin cell and boost converter, the power rail is neither very precise nor very stable.

The STM32L031 includes an on-chip bandgap reference with a nominal 1.2V value and ±100ppm/°C tempco.

At manufacture, the value of this internal reference is read by the ADC while the chip is supplied with a precise supply voltage (3.0 ±0.01V). The resulting code is burned into non-volatile memory on the chip. You can use this to convert a supply-referenced ADC reading to an absolute voltage with known precision.

The ADC also includes an auto-calibration feature that automatically performs a zero calibration. This only works to eliminate on-chip offset errors, such as those related to the analog multiplexer, the sample-and-hold system and the ADC itself. If you want to eliminate off-chip offset errors, you need to provide the hardware and do this yourself, as we discussed in the third article in this series.

In addition, the ADC includes an oversampler that automatically makes several sequential conversions (up to 256), sums the result, then scales the result back by some factor to get an averaged result. Of course, you could do this in firmware, but the hardware oversampler does everything in the background for you.

This technique can actually improve the precision of ADC measurements in the presence of noise. This means that a 12-bit ADC could appear to have 13 or more bits of precision.

The reason this works is shown in Fig.9. Here, we have a noisy signal with an average value between the nth and nth+1 thresholds of an ADC. If we were to take just one sample, we could get either result. In fact, it is possible we could get a result one more bit higher and lower if we are unlucky with our sampling.

If we make many measurements, however, some will be high and some low, but their average will lie somewhere between the two thresholds. We could say the resulting measurement is at the nth+½ threshold, effectively giving us an extra bit of resolution.

This only works if the noise has an average value of zero and is uncorrelated with the sampling rate, and the noise has to have sufficient magnitude. Sometimes, a designer will deliberately introduce noise or some other form of 'dither' to a signal to increase the resolution when oversampling.

Design decisions

The relevant part of the test circuit I built is shown in Fig.10. The challenge is to digitise a bipolar (in this case, ±6V) signal with a single-ended ADC and a single 3.3V supply.

The input signal is coupled to the ADC by a difference amplifier with a gain of 0.25, reducing the 12V input span to 3V, within the ADC input range. Using a difference amp here allows the input voltage to extend well beyond the supply rails without getting into problems with an op amp's common-mode input range.

The reference input of the difference amplifier is connected to the mid-point of the power supply derived from a voltage divider and buffer op amp. The output of the difference amplifier will therefore $V_{OUT} = 0.25 (V_{IN}+ - V_{IN}-) + 1.65V$.

This means the voltage applied to the ADC will be in the range 1.65 ±1.5V (0.1V to 3.2V) over the ±6V input span,

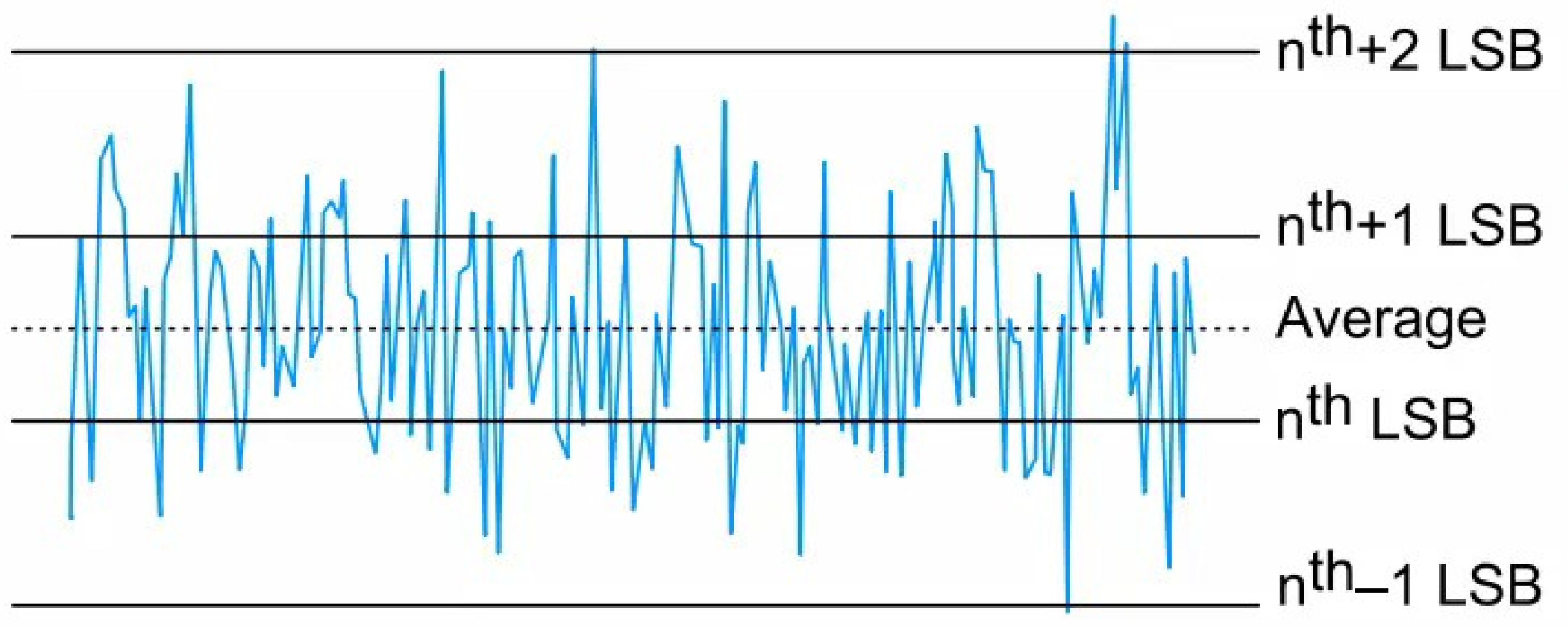


Fig.9: it is possible to increase the effective resolution of an ADC for slowly changing signals by averaging many samples of a noisy signal.

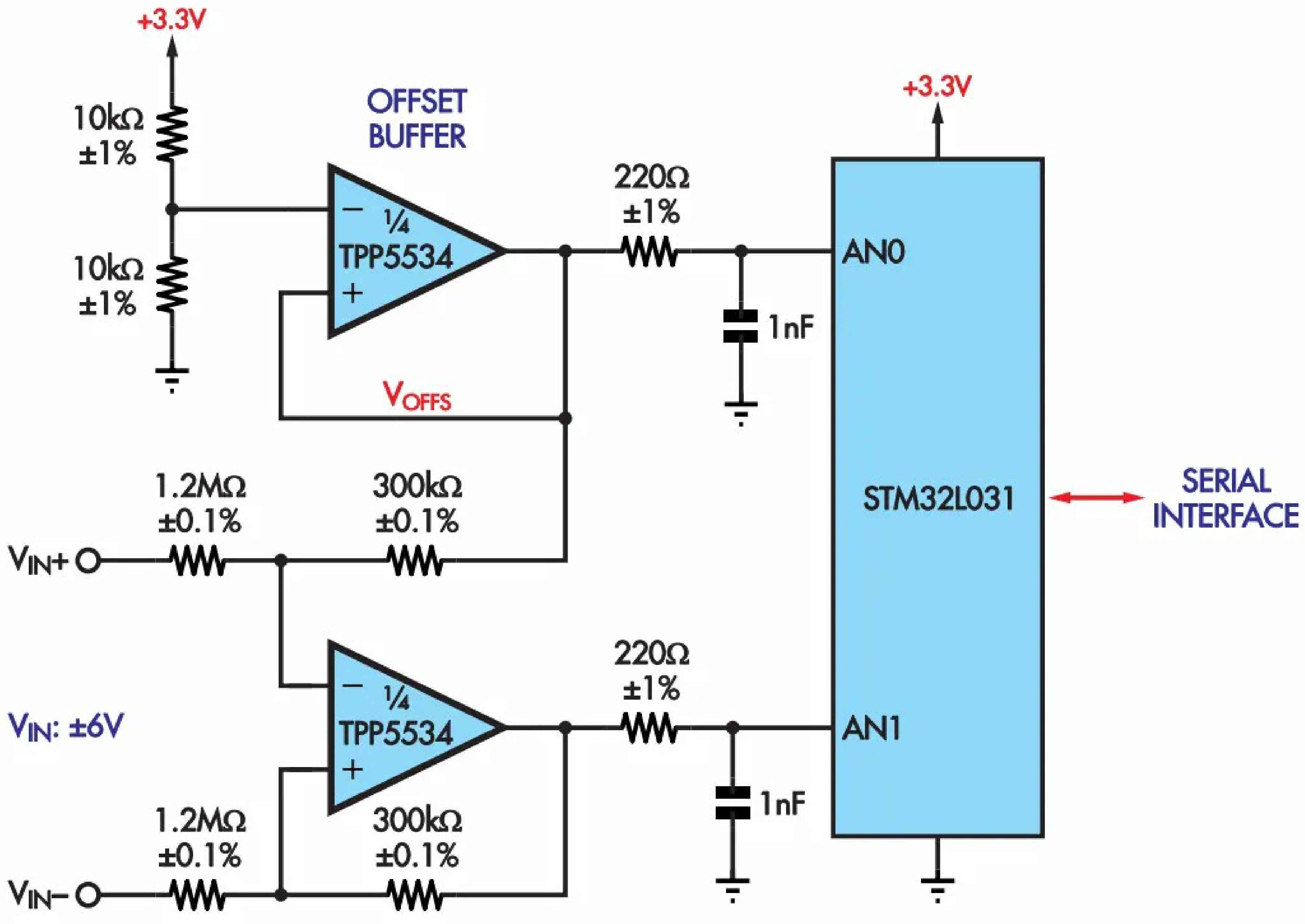


Fig.10: the test circuit digitises a ±6V input using a difference amplifier with a gain of 0.25 and an offset of about 1.65V. The latter value is not critical, since this voltage is also digitised and the measured value is used to reconstruct the input voltage.

avoiding the ends of the ADC input range near the supply rails where we know errors may lie. I have used low-cost TP5534 zero-drift op amps and 0.1% gain setting resistors to keep the analog errors down.

The error budget table (Table 1) shows the analog error (line 5) is around ±0.2% in the worst case – almost all down to the gain resistor tolerance. The offset errors in lines 1 and 2 of the table are very low, so the internal zero calibration should be sufficient for our purposes. It is probably not worth using higher precision resistors here, since the ADC and calibration errors are of a similar magnitude.

The ADC TUE of ±4LSB corresponds to a relative error of just over 0.1% and an absolute error of ±3.2mV. The relative error is easy to calculate from the TUE and the ADC precision: $100\% \times \text{TUE} \div (2^n - 1)$, where n is the ADC precision in bits.

We don't need to use precision resistors to create the mid-supply voltage since we also digitise this voltage and subtract it in firmware. The absolute value of the mid-point voltage therefore does not matter – it just has to be close to half the supply voltage.

I set the ADC up with an 8MHz clock giving a cycle time of 125ns. A single conversion consists of a sampling period, which is programmable, and a conversion time of 12.5 clock cycles.

I chose a sampling time of 19.5 cycles (about 2.5µs) to be 10 or more times longer than the time constant of the external RC filter, and that of the internal filter made up of the 8pF sampling capacitor and the 1kΩ resistance of the analog multiplexer and sample-and-hold switches.

I configured the oversampler to take 256 samples and to divide the resulting sum by 256 to restore 12 bits of resolution. There is no point in going for higher resolution, since the analog errors and the ADC TUE are already at this level of precision. No amount of oversampling will compensate for errors that affect every sample to the same extent.

We use the same oversampling on all three ADC conversions: the main input, the mid-supply offset and the internal voltage reference. I also performed an internal zero-calibration on initialising the ADC to make sure any offset errors in the input multiplexer and sample and hold were minimised.

Scaling the ADC results

The relationship between the ADC code N and the absolute voltage on an ADC pin is $V_{IN} = V_{DD}\,(N \div 4095)$, where V_{DD} is the microcontroller's power supply voltage. We don't know this voltage precisely, but we can work it out by using the ADC to read the internal reference (NIREF) and the stored ADC code (NCAL) that was converted with a known supply voltage:

$$\mathbf{V_{DD} = 3.0 \times (NCAL \div NIREF)}$$

Having calculated the absolute voltage of the input and the midpoint offset voltage (in millivolts, since we are dealing with integers), we can use these, plus the nominal differential amp gain, to calculate the overall circuit input voltage.

I did all of this and measured the input voltage, the ADC channel input voltage and read out the digital result. The results are pretty impressive – the measured error is better than ±0.05%, corresponding to ±3mV on the full-scale ±6V input range. This is 1 part in 2000, or about 11 bits of effective resolution.

The transfer function of the ADC turned out to be $f(x) = 0.9995x - 0.0002$ with an R^2 of 1.000. The line-of-best-fit gain error is less than ±0.01% and the offset error is less than 1mV. The worst individual sample error was better than ±0.05%.

These results are an order of magnitude better than the 0.6% error calculated in the error budget. To some extent, this is to be expected (the odds are low that we will have the worst-case errors everywhere), but it is worth a bit of a closer look at why it performs better than expected.

First, the calculated analog gain error is almost all due to the resistor tolerances, which would have to all be at the extremes of their tolerance band – that is unlikely in practice (but possible, of course).

Second, the calculated TUE includes offset error, which is nulled out through the zero-calibration process. And finally, the 0.3% error on the supply voltage for the factory calibration seems to me to be a very conservative figure. I would be surprised if the supply voltage was not regulated more tightly than ±10mV during this step, so will probably be better than specified by an order of magnitude.

So, in conclusion, the typical 10- to 12-bit ADCs used in microcontrollers are really useful, but they have some limitations – especially if you are using the power supply voltage as the reference. Read the data carefully, since they will likely have fewer bits of effective resolution once the TUE is taken into account.

To most effectively use the bits at your disposal, you should think seriously about averaging many samples if your microcontroller has the time to do so. That depends on how quickly your input is changing, how fast your ADC is, how many measurements you have to make, how often and so on.

As I have mentioned before, in precision applications, you should also avoid using the very ends of the ADC span. PE

	Error Source (25˚C)	Nominal Value	Absolute Error	Relative Error
1	Op Amp Offset Voltage (±20µV, 0.05µV/˚C)	0V	20µV	0.000%
2	Voltage due to Op Amp Offset Current (±100pA, 300kΩ || 1.2MΩ)	0V	24µV	0.000%
3	Total error at Op Amp Input (Line 1 + Line 2)	0V	44µV	0.001%
4	Op Amp Gain Error (0.1% resistors)	1.25		0.200%
5	Voltage error at ADC Input (Line 3 × Line 4)	0V	12mV	0.201%
6	ADC (TUE ±4LSB – least significant bits)	0V	3.2mV	0.107%
7	Total error in ADC code (Line 5 × Line 6)	0V	10.2mV	0.308%
8	Error in internal V_{REF} calibration (@3.0±0.01V)	0V	10mV	0.303%
9	Total error (Line 7 × Line 8)	0V	20.2mV	0.611%

ETI BUNDLE (1) Teach-In 3, 4 and 5 – all on CD-ROM – only £18.95

ELECTRONICS TEACH-IN 3 – CD-ROM

Mike & Richard Tooley

ELECTRONICS TEACH-IN 3
CIRCUIT SURGERY
PRACTICALLY SPEAKING
INGENUITY UNLIMITED
FREE TEACH-IN 1 CD-ROM
TWO TEACH-INs FOR THE PRICE OF ONE!

The three sections of the *Teach-In 3* CD-ROM cover a huge range of subjects that will interest everyone involved in electronics – from newcomers to the hobby and students to experienced constructors and professionals.

The first section (80 pages) is dedicated to *Circuit Surgery*, *EPE/PE's* regular clinic dealing with readers' queries on circuit design problems – from voltage regulation to using SPICE circuit simulation software.

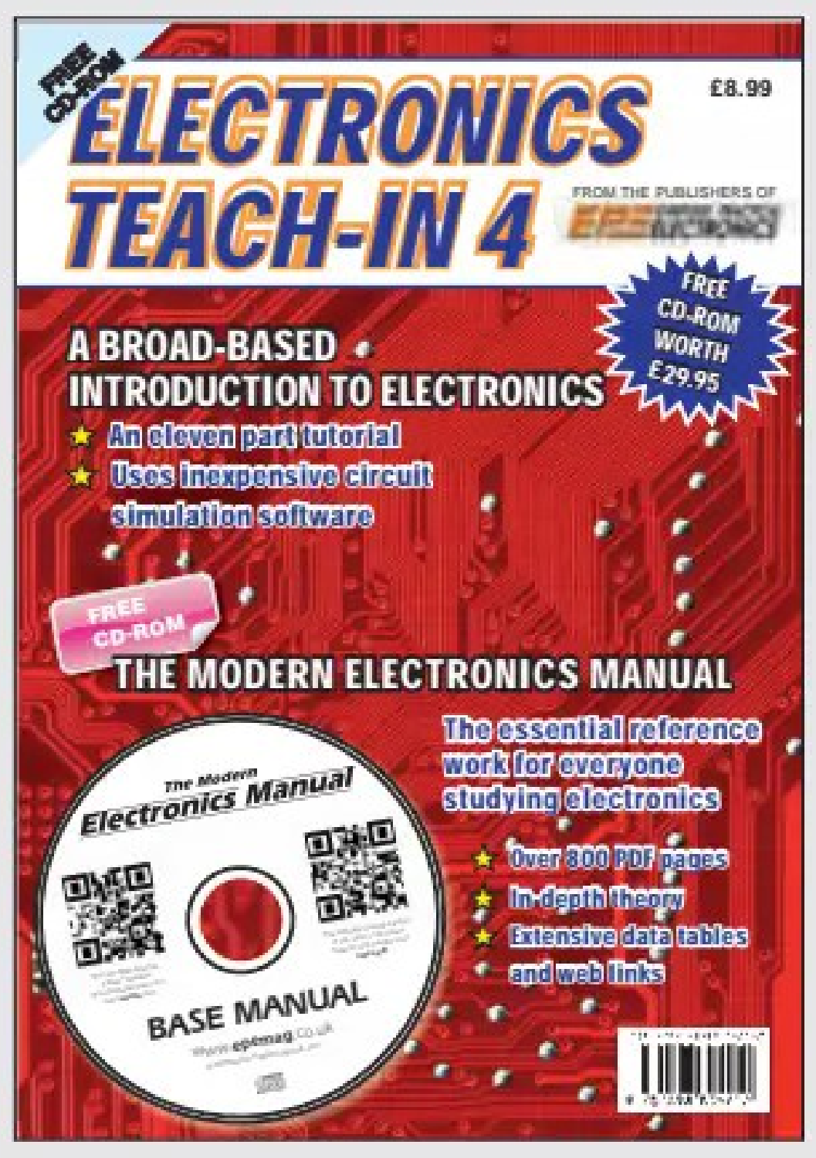
The second section – *Practically Speaking* – covers hands-on aspects of electronics construction. Again, a whole range of subjects, from soldering to avoiding problems with static electricity and identifying components is covered. Finally, our collection of *Ingenuity Unlimited* circuits provides over 40 circuit designs submitted by readers.

The CD-ROM also contains the complete *Electronics Teach-In 1* book, which provides a broad-based introduction to electronics in PDF form, plus interactive quizzes to test your knowledge and TINA circuit simulation software (a limited version – plus a specially written TINA Tutorial).

The *Teach-In 1* series covers everything from electric current through to microprocessors and microcontrollers, and each part includes demonstration circuits to build on breadboards or to simulate on your PC.

ELECTRONICS TEACH-IN 4 – CD-ROM
A BROAD-BASED INTRODUCTION TO ELECTRONICS

Mike & Richard Tooley


ELECTRONICS TEACH-IN 4
A BROAD-BASED INTRODUCTION TO ELECTRONICS
THE MODERN ELECTRONICS MANUAL

The *Teach-In 4* CD-ROM covers three of the most important electronics units that are currently studied in many schools and colleges. These include, Edexcel BTEC level 2 awards and the electronics units of the Diploma in Engineering, Level 2.

The CD-ROM also contains the full *Modern Electronics Manual*, worth £29.95. **The Manual** contains over 800 pages of electronics theory, projects, data, assembly instructions and web links.

A package of exceptional value that will appeal to anyone interested in learning about electronics – hobbyists, students or professionals.

ELECTRONICS TEACH-IN 5 – CD-ROM
JUMP START

Mike & Richard Tooley

ELECTRONICS TEACH-IN 5
JUMP START
PRACTICALLY SPEAKING
PIC 'N' MIX
TEACH-IN 2

15 design and build circuit projects for newcomers or those following courses in school and colleges.

The projects are: ■ Moisture Detector ■ Quiz Machine ■ Battery Voltage Checker ■ Solar-Powered Charger ■ Versatile Theft Alarm ■ Spooky Circuits ■ Frost Alarm ■ Mini Christmas Lights ■ iPod Speaker ■ Logic Probe ■ DC Motor Controller ■ Egg Timer ■ Signal Injector Probe ■ Simple Radio Receiver ■ Temperature Alarm.

PLUS

PIC'n' Mix – starting out with the popular range of PIC microcontrollers and *Practically Speaking* – tips and techniques for project construction.

The CD-ROM also contains:

- ■ Complete *Teach-In 2* book, a practical introduction to PIC microprocessors
- ■ MikroElektronika, Microchip and L-Tek PoScope software.

Three Teach-ins for the great price of £18.95

PLUS you also get the contents of the free CD-ROM from each issue...

... so that's another TWO Teach-Ins and The Full Modern Electronics Manual!

What a Bargain!!

ETI BUNDLE (2) Teach-In 6, 7 and 8 – all on CD-ROM – only £18.95

ELECTRONICS TEACH-IN 6 – CD-ROM
A COMPREHENSIVE GUIDE TO RASPBERRY Pi

Mike & Richard Tooley

Teach-In 6 contains an exciting series of articles that provides a complete introduction to the Raspberry Pi, the low cost computer that has taken the education and computing world by storm.

This latest book in our *Teach-In* series will appeal to electronics enthusiasts and computer buffs who want to get to grips with the Raspberry Pi.

Teach-In 6 is for anyone searching for ideas to use their Pi, or who has an idea for a project but doesn't know how to turn it into reality. This book will prove invaluable for anyone fascinated by the revolutionary Pi. It covers:

- ■ Pi programming
- ■ Pi hardware
- ■ Pi communications
- ■ Pi Projects
- ■ Pi Class
- ■ Python Quickstart
- ■ Pi World
- ■ ...and much more!

The *Teach-In 6* CD-ROM also contains all the necessary software for the series, so that readers and circuit designers can get started quickly and easily with the projects and ideas covered.

ELECTRONICS TEACH-IN 6
RASPBERRY Pi
A COMPREHENSIVE GUIDE TO RASPBERRY Pi

ELECTRONICS TEACH-IN 7 – CD-ROM
DISCRETE LINEAR CIRCUIT DESIGN

Mike & Richard Tooley

Teach-In 7 is a complete introduction to the design of analogue electronic circuits. It is ideal for everyone interested in electronics as a hobby and for those studying technology at schools and colleges. The CD-ROM also contains all the circuit software for the course, plus demo CAD software for use with the *Teach-In* series.

- ■ Discrete Linear Circuit Design
- ■ Understand linear circuit design
- ■ Learn with 'TINA' – modern CAD software
- ■ Design simple, but elegant circuits
- ■ Five projects to build:
  - i) Pre-amp
  - ii) Headphone Amp
  - iii) Tone Control
  - iv) VU-meter
  - v) High Performance Audio Power Amp.

PLUS

Audio Out – an analogue expert's take on specialist circuits

Practically Speaking – the techniques of project building.

ELECTRONICS TEACH-IN 7
DISCRETE LINEAR CIRCUIT DESIGN

ELECTRONICS TEACH-IN 8 – CD-ROM
INTRODUCING THE ARDUINO

Mike & Richard Tooley

Hardware: learn about components and circuits
Programming: powerful integrated development system
Microcontrollers: understand control operations
Communications: connect to PCs and other Arduinos.

Teach-In 8 is an exciting series designed for electronics enthusiasts who want to get to grips with the inexpensive, popular Arduino microcontroller, as well as coding enthusiasts who want to explore hardware and interfacing. It will provide a one-stop source of ideas and practical information.

The Arduino offers a truly effective platform for developing a huge variety of projects; from operating a set of Christmas tree lights to remotely controlling a robotic vehicle through wireless or the Internet.

Teach-In 8 is based around a series of practical projects with plenty of information for customisation.

This book also includes *PIC n' Mix*: 'PICs and the PICkit 3 – A Beginners guide' by Mike O'Keefe and *Circuit Surgery* by Ian Bell – 'State Machines part 1 and 2'.

The CD-ROM includes the files for:

- ■ *Teach-In 8*
- ■ Microchip MPLAB IDE XC8 8-bit compiler
- ■ PICkit 3 User Guide
- ■ Lab-Nation Smartscope software.

ELECTRONICS TEACH-IN 8
INTRODUCING THE ARDUINO

ORDER YOUR BUNDLE TODAY!

JUST CALL 01202 880299 – OR VISIT www.electronpublishing.com

PROJECT BY JOHN CLARKE

REPURPOSING THE MAINS POWER-UP SEQUENCER

SILICON CHIP Mains Power-Up Sequencer
Mains Power Input
Fuse 10A
OUT1
OUT3
OUT4
Power
1
2
3
4

Generators and inverters are not always powerful enough to run more than one high-current appliance at a time. For example, if you have more than one refrigerator or a separate freezer and fridge and want to run them off-grid, they may need to run at different times. The Mains Power-Up Sequencer from the February & March issues can be programmed to do that automatically.

The Mains Power-Up Sequencer from the February & March issues earlier this year was intended for powering up appliances in sequence with brief delays in between to avoid overloading a circuit breaker at switch-on.

However, the fact that each outlet is controlled independently by a microcontroller means that the way each outlet is controlled can be changed with new software.

We hadn't considered this second application until a reader wrote to us. In part, he wrote:

Say a business has several fridges/freezers to run from a small emergency power source. It would be very useful to be able to sequence the output to several loads for varying periods, like 15 or 20 minutes, making it unnecessary to manually switch loads to avoid overloading a generator or inverter.

A shortlist of features

- Powers on two to four mains outlets individually in a rotating sequence
- Adjustable powered-on period of eight seconds to 30 minutes
- Optional daisy-chain connection for up to four more outlets (up to eight total)
- 'Phantom Appliance' load detection option (for up to four outlets)

Happily, we can satisfy this request. The re-purposed version of the project mainly requires the microcontroller software to be changed, plus some minor wiring adjustments.

Three options

Three new versions of the Sequencer are described here, all using the same revised software. The first is called the Primary unit (see Fig.1). It operates with a rotating sequence, switching on Outlet 1 for a period, then switching it off before switching on Outlet 2 for the same period. This sequence continues for all outlets, and when Outlet 4 switches off, the sequence repeats.

The power-on period is adjustable from eight seconds to 30 minutes. The eight-second period is mainly useful for testing the unit to see if it works without waiting too long. As with the original Sequencer, you can build it to have fewer than four outlets. If only two or three are required, it will return to Outlet 1 after Outlet 2 or Outlet 3 switches off.

The second version operates similarly to the Primary version but includes current detection. When an outlet is first powered, it monitors the current drawn. If an appliance draws power, the outlet stays powered. The outlet switches off after the timeout period, or earlier if the appliance draws less than 35W.

We call this the Phantom Appliance Detection (PAD) mode, where only the outlets that have an appliance connected (or are ready to run in the case of a fridge or freezer) will be included in the sequence.

This mode can be useful for powering refrigerators and freezers because they don't run constantly. Powering an outlet for an appliance that is not doing anything useful wastes time, since it could power the next fridge or freezer instead. Also, the fridge or freezer may finish running its compressor before the timeout expiry. In this case, the PAD unit will move on to power the next appliance early.

This mode is also useful where you have the four outlets on the Sequencer, but you may sometimes only use it for two or three appliances. The Sequencer will skip over the unused outlets, and you won't have to manually change the configuration to set the number of outlets used. It also gives you the flexibility to switch one or more loads off when you want them to be skipped.

Daisy-chain mode

The third configuration, Daisy Chain, can give you more than four outlets (up to eight). Daisy-chaining is impractical for PAD units; only the Primary unit can be daisy-chained. That is because the AN4 input of microcontroller IC9 used to enable daisy chaining is also

Repurposing the Mains Sequencer

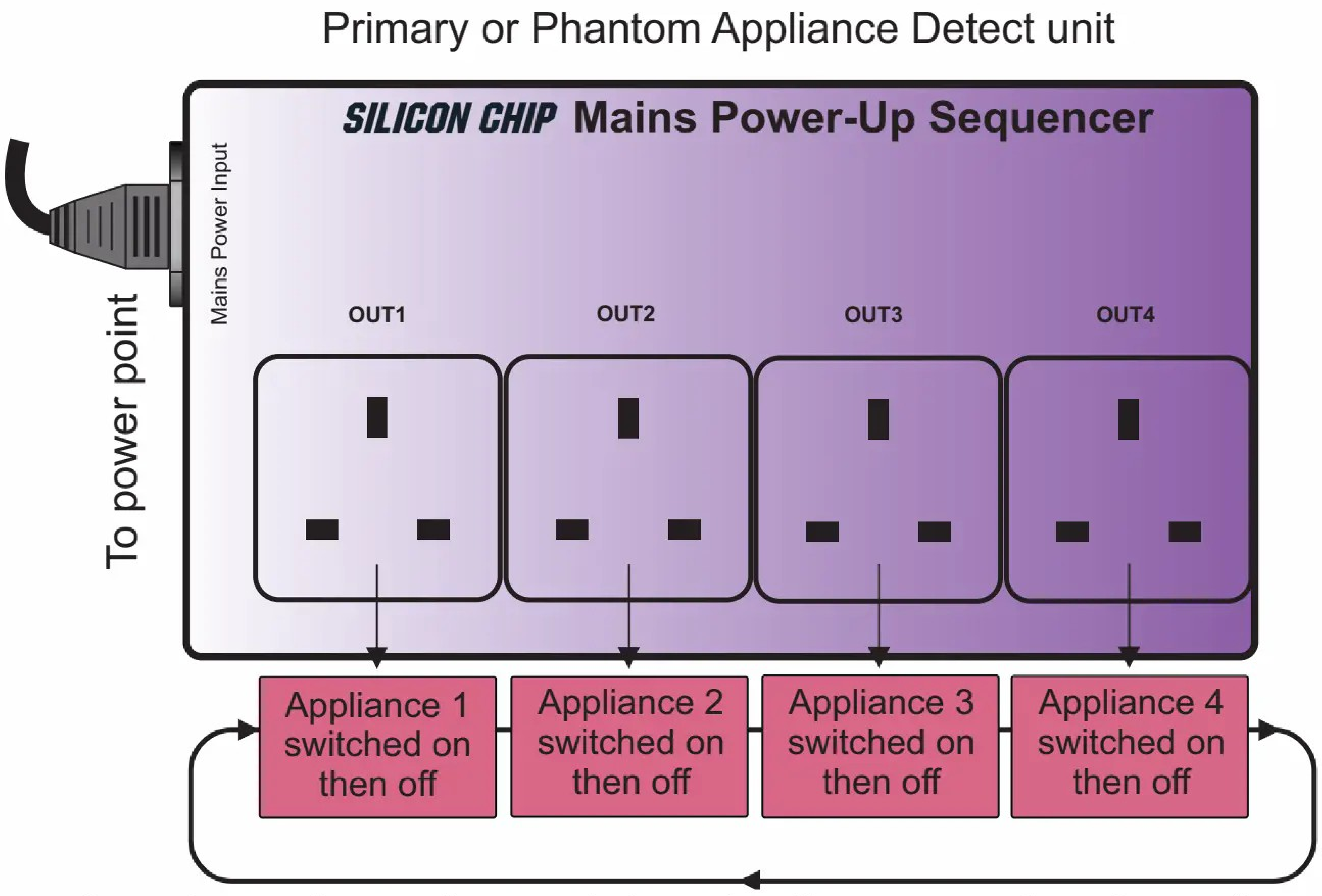

Primary or Phantom Appliance Detect unit
SILICON CHIP Mains Power-Up Sequencer
To power point
Mains Power Input
OUT1
OUT2
OUT3
OUT4
Appliance 1 switched on then off
Appliance 2 switched on then off
Appliance 3 switched on then off
Appliance 4 switched on then off

Fig.1: when used as a Primary unit, it continually sequences through up to four outlets, switching them on for a fixed time in turn. The PAD unit is similar, except it will only switch on outlets with an appliance connected and drawing at least 35W. Otherwise, after a 1s delay, it skips that outlet.

used for current detection on a PAD unit, and it can't perform both jobs simultaneously.

Fig.2 shows how a Daisy Chain unit is connected to the Primary unit. The Daisy Chain unit monitors the last outlet from the Primary unit via its Mains Detect Input. Its Outlet 1 is powered after the last outlet from the primary unit (shown as Outlet 4) powers on and then off. The Daisy Chain unit then powers each outlet on and off in sequence, stopping after the last outlet.

When used in this mode, the Primary unit powers each outlet on and off in sequence, but after powering Outlet 4 off, there is a delay before powering Outlet 1 again. That gives the Daisy Chain unit time to run its complete sequence. We call this delay the return delay, and it is set so that the Daisy Chain unit finishes its entire cycle before the Primary unit starts the cycle again.

The return delay can be selected as between one to four times the usual delay period that is set with VR1. That allows you to build the Daisy Chain unit with between one and four additional outlets, with the delay multiplier on the Primary unit set to match the number of outlets on the Daisy Chain unit.

Version selection

The three versions use the same software but run different sections of the code. The software checks the RA3 and AN4 inputs to the microcontroller at power-up.

If the RA3 input is low (near 0V), the software runs for a PAD unit, while if RA3 is high (5.1V), either the Primary or Daisy Chain code runs. To discern between these two options, it checks the AN4 input. If the voltage is low, the software determines it is a Daisy Chain unit. If the AN4 input is above the threshold voltage for mains voltage detection, it runs the code for a Primary unit.

It is important to build the Sequencer according to the build details for the version you are making so the software runs correctly.

Building it

This article mainly describes the changes required for the new functions, so for the full PCB assembly instructions, you will need to refer to the articles in the February & March 2025 issues.

Those articles describe various build options. You can build a unit with between one and four mains outlets (see Table 1), and the optional Current Detection and Mains Voltage Detection circuitry may need to be included.

With the new software, switches S1-S3 provide functions different from the original Mains Power-Up Sequencer, as shown in Table 2 and Table 3. VR1 is now only used to adjust the power-on period for each outlet.

The wiring and PCB changes for all three versions are shown in Fig.3. In all cases, the two connections at CON7 are bridged using 10A mains wire.

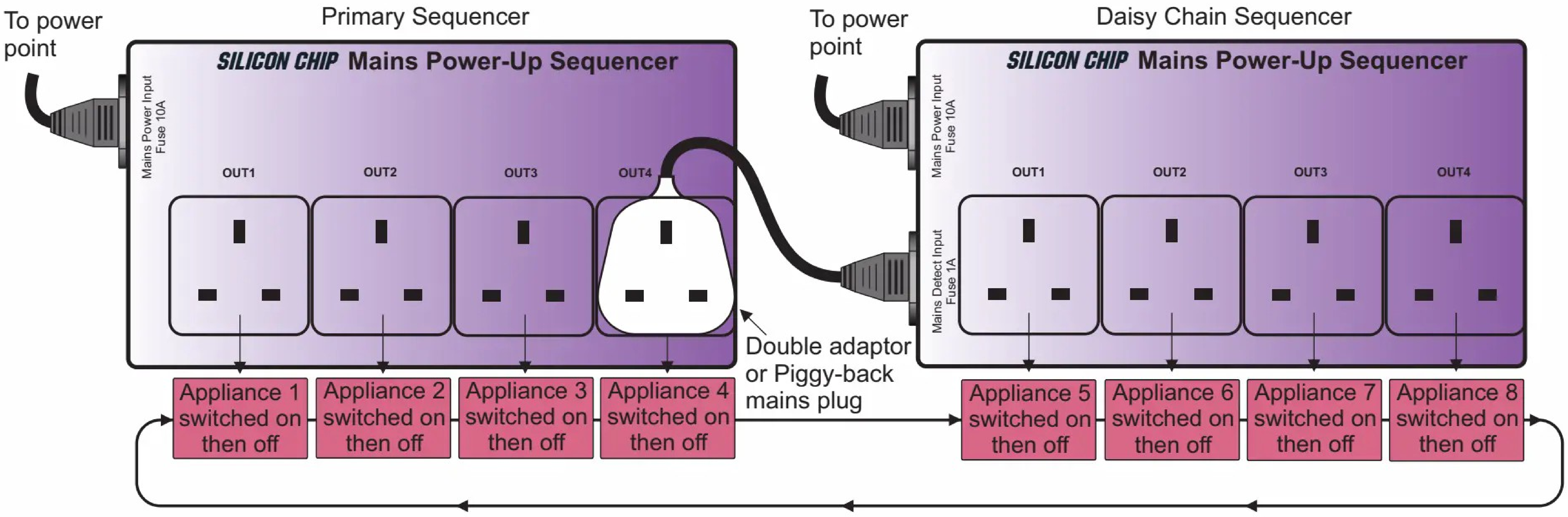
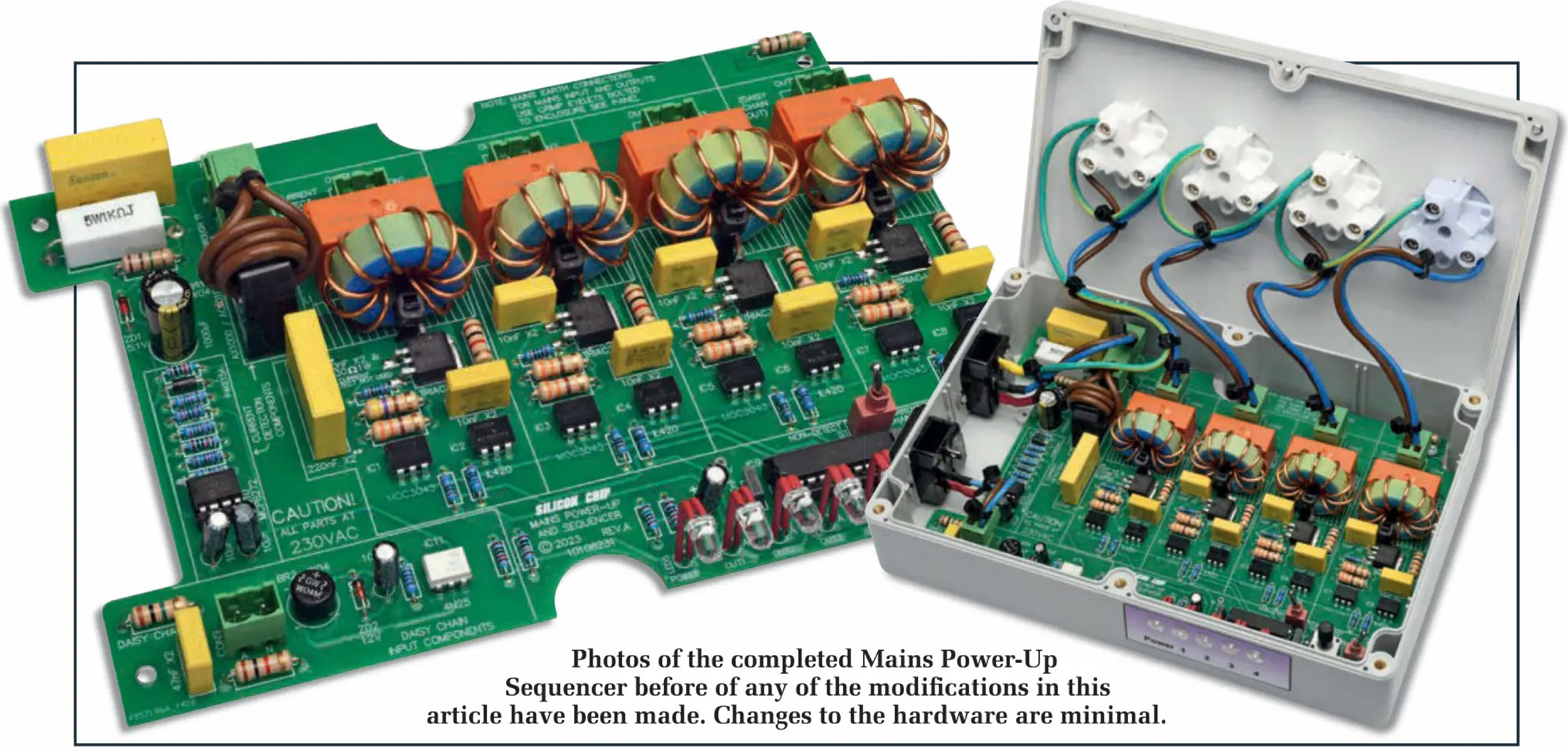

To power point
Primary Sequencer
SILICON CHIP Mains Power-Up Sequencer
Mains Power Input Fuse 10A
OUT1
OUT2
OUT3
OUT4
Appliance 1 switched on then off
Appliance 2 switched on then off
Appliance 3 switched on then off
Appliance 4 switched on then off
Double adaptor or Piggy-back mains plug
To power point
Daisy Chain Sequencer
SILICON CHIP Mains Power-Up Sequencer
Mains Power Input Fuse 10A
Mains Detect Input Fuse 1A
OUT1
OUT2
OUT3
OUT4
Appliance 5 switched on then off
Appliance 6 switched on then off
Appliance 7 switched on then off
Appliance 8 switched on then off

Fig.2: for more than four outlets (up to eight), you can connect a Daisy Chain unit to a Primary unit, as shown here. The Daisy Chain unit is triggered when the last Primary outlet switches off; the Primary unit waits for the Daisy Chain unit to finish before restarting the sequence.



Photos of the completed Mains Power-Up Sequencer before of any of the modifications in this article have been made. Changes to the hardware are minimal.

Current transformer T1 is only used for the PAD unit, with the mains Active wire passing through T1's core.

The snubber components for the OUT1 circuitry across TRIAC1 are a 10nF X2-rated capacitor for C1 and a 330Ω 1W resistor for R1. Do not use the alternative 220nF X2 rated capacitor and 470Ω 1W resistor values mentioned in the original articles.

Microcontroller IC9 must be programmed with the revised software, coded 1010823M.hex. You can download the HEX file and assembly language source code (siliconchip.au/Shop/6/358) and program the chip using a PIC programmer. Or you can purchase a programmed microcontroller from the Silicon Chip website.

The above components and wiring changes are common to all the revised versions, but specific modifications are required for each version, as described below.

Primary unit

For the Primary unit, the current and voltage detection sections are left unpopulated. The mains Active wire shown going through the current transformer for the original Sequencer instead goes directly to CON6. Place a wire link between pins 4 and 5 of the pads for IC11 so that the Sequencer will run the Primary unit version of the code at start-up.

You can select the number of active outlets by making the linking options as shown in Table 1. Set VR1 for the required on-period of each outlet. Fully clockwise sets a 30-minute timeout; a mid setting is about 15 minutes.

If the Primary unit is not being used with a Daisy Chain unit, set switch S3's lever to the left, so there is no return delay (see Table 2). The switch positions for S1 and S2 do not matter for this version.

If the Primary unit is used with a Daisy Chain unit, set switch S3's lever right so there is a return delay (see Table 2). The return delay setting is made using switches S1 & S2, as shown in Table 3; select the ×4, ×3, ×2 or ×1 delay multiplier to match the number of outlets used on the Daisy Chain unit.

Daisy Chain unit

The Daisy Chain unit requires the voltage detection circuitry to be installed, with no wire link between pins 4 & 5 of IC11's pads. You can select the number of outlets installed in the Daisy Chain unit as per Table 1. Set VR1 for the required power-on period of each outlet, but ensure it is slightly less than the period set for the Primary unit.

Set switch S3 for the Daisy Chain version with the lever to the left so there is no return delay (see Table 2). The switch lever positions for S1 and S2 do not matter for this version.

Phantom Appliance Detect (PAD) unit

The PAD version requires the current detection circuitry to be installed, with the Active wire from CON5 looping through current transformer T1 before terminating at CON6.

You will also need to connect a wire between the 0V test point and the bottom of the 10kΩ resistor that con-

Table 1 – setting the number of outlets (for all units)

# outlets	RA1 (pin 18)	RA0 (pin 19)
4 (default)	0V (PCB bottom layer)	0V (PCB bottom layer)
3	0V (PCB bottom layer)	5.1V (PCB top layer)
2	5.1V (PCB top layer)	0V (PCB bottom layer)
1	5.1V (PCB top layer)	5.1V (PCB top layer)

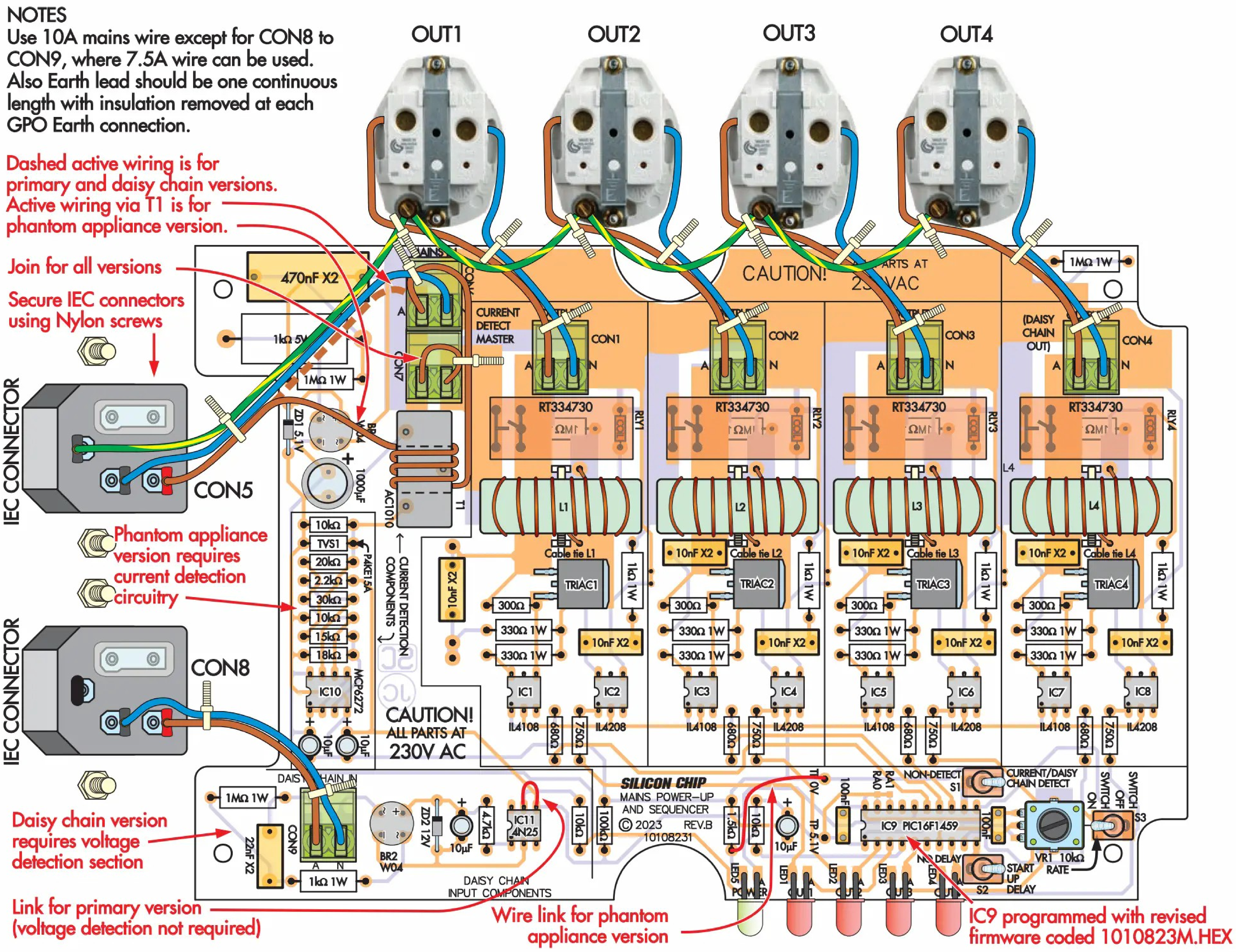
Table 2 – return delay setting

Return delay	S3 position
No	Left (open)
Yes	Right (closed)

Table 3 – return delay multiplier (for daisy-chained Primary unit)

Multi.	S1 position	S2 position
×4	Left (open)	Left (open)
×3	Right (closed)	Left (open)
×2	Left (open)	Right (closed)
×1	Right (closed)	Right (closed)

Repurposing the Mains Sequencer


NOTES
Use 10A mains wire except for CON8 to CON9, where 7.5A wire can be used. Also Earth lead should be one continuous length with insulation removed at each GPO Earth connection.

Dashed active wiring is for primary and daisy chain versions. Active wiring via T1 is for phantom appliance version.

Join for all versions

Secure IEC connectors using Nylon screws

Phantom appliance version requires current detection circuitry

Daisy chain version requires voltage detection section

Link for primary version (voltage detection not required)

Wire link for phantom appliance version

IC9 programmed with revised firmware coded 1010823M.HEX

OUT1
OUT2
OUT3
OUT4
IEC CONNECTOR
CON5
IEC CONNECTOR
CON8
470nF X2
1kΩ 5W
1MΩ 1W
ZD1 5.1V
1000µF
BR1 W04
AC1010
T1
CON6
CON7
CURRENT DETECT MASTER
CURRENT DETECTION COMPONENTS
10kΩ
TVS1
20kΩ
2.2kΩ
30kΩ
10kΩ
15kΩ
18kΩ
P4KE15A
IC10
MCP6272
10µF
10µF
10nF X2
CAUTION! ALL PARTS AT 230V AC
CAUTION! ALL PARTS AT 230VAC
CON1
CON2
CON3
CON4
(DAISY CHAIN OUT)
A
N
RT334730
RLY1
RLY2
RLY3
RLY4
L1
L2
L3
L4
Cable tie L1
Cable tie L2
Cable tie L3
Cable tie L4
TRIAC1
TRIAC2
TRIAC3
TRIAC4
1kΩ 1W
10nF X2
300Ω
330Ω 1W
330Ω 1W
IC1
IC2
IC3
IC4
IC5
IC6
IC7
IC8
IL4108
IL4208
680Ω
750Ω
1MΩ 1W
DAISY CHAIN IN
CON9
22nF X2
1kΩ 1W
BR2 W04
ZD2 12V
10µF
4.7kΩ
IC11 4N25
10kΩ
100kΩ
DAISY CHAIN INPUT COMPONENTS
SILICON CHIP
MAINS POWER-UP AND SEQUENCER
© 2023 REV.B
10108231
1.5kΩ
10kΩ
10µF
TP 0V
TP 5.1V
100nF
RA1
RA0
NON-DETECT
S1
CURRENT/DAISY CHAIN DETECT
IC9 PIC16F1459
100nF
VR1 10kΩ
SWITCH ON
SWITCH OFF
S3
NO DELAY
START UP DELAY
S2
RATE
LED5 POWER
LED1 OUT1
LED2 OUT2
LED3 OUT3
LED4 OUT4

Fig.3: besides reprogramming IC9 with the new software, just a few changes are required to the hardware. Add one of the wire links shown in red if building the Primary or PAD version. The mains Active wire only goes through T1 for the PAD version; otherwise, it connects directly to CON6.

nects (through tracks on the PCB) to the pin 4 RA3 input of IC9, as shown above. This informs the software that the unit is the PAD version. With the revised software, the RA3 pin is set as a digital input rather than as a master clear ($\overline{\text{MCLR}}$) reset line, as it was in the original version of the Sequencer.

Put switch S3's lever left so there is no return delay (see Table 2). The switch lever positions for S1 and S2 do not matter for this unit.

Testing

As per the original Mains Power-Up Sequencer articles mentioned, all wiring and adjustments must be made with the input mains power disconnected since the circuitry is live when plugged in. Also, any adjustments of the period using VR1 are only detected at power up, so there is no benefit to adjusting VR1 with the power on.

So, each time you want to make an adjustment, ensure that the power is off before opening the lid of the enclosure. Replace the enclosure lid before reapplying power.

If VR1 is set to its minimum fully anti-clockwise position, the period for each outlet will be short, at eight seconds. That makes monitoring and checking its operation easier. You can see the sequencing occur as the indicator LEDs light up for each outlet.

For the PAD unit, the output LED indicator for each channel will only light when an appliance that draws power (at least 35W) is detected. That is because only the Triac for each channel is switched on initially, while the current drawn by the appliance is first detected, and the LED indicators only show the relay status.

Using the Triac to apply voltage initially saves the relay from operating if there is no current drawn by the appliance, extending the relay life considerably.

For the PAD sequence, you can check each outlet by connecting a load that will draw 35W or more, such as a 40W 230V AC halogen lamp. Cycling through outlets that do not have a load connected takes one second per outlet. PE

Warning: Mains Voltage

All circuitry within the Mains Sequencer operates at Line (mains) voltages. It would be an electrocution hazard if built incorrectly or used with the lid open. Only build this if you are fully experienced in building mains projects.

AOShop

The home for specialist audio, analogue and historic components – provided by Jake Rothman, *PE's Audio Out* columnist. The AOShop is your best bet for classic analogue 'NOS' (new old stock) components, including all parts for *Audio Out* projects and designs.

Ge semiconductors

Small-signal PNP transistors

NKT214F, OC57, OC59, 2N1377, 2N525	£1.00

Low-noise PNP transistors

GET106	£2.50

Small power PNP transistors

AC153, AC153K, AC188	£1.50

Small power NPN transistors

AC176, AC141K	£2.00

High-voltage PNP transistors

OC77, CV7001	£1.50

TO3 PNP power transistors

OC22, CV7054 (OC23), OC25, OC35, OC36, AD143, AD149, AD161, AD162	£2.00
AD140	£3.50
AD149, AD161/2 matched pair	£5.00

RF PNP transistors

OC41, OC42, 2SA12, 2SA53,AF124, AF178, GET872A	£1.50

NPN transistors

OC139, OC140, ASY73	£2.00

Diodes

CV7049 (OA10), CG92 (OA91)	£0.50

Si semiconductors

Diodes

ZC5800 RF Schottky	£0.20

Low-noise silicon transistors

BFW16A, 2SC3071, 2SC3068, 2SA1016K, 2SC2362K, 2SA970BL	£1.50
2SC2204, 2SD655, BC550C	£0.50
ZTX651	£0.30
ZTX751	£0.50

RF transistors (suitable for Theremin)

BF199	£0.50

Audio power MOSFETs

Exicon 10N20, 10P20	£6.50
Hitachi 2SJ99, 2SK343	£3.50
Hitachi 2SJ56, 2SK176	£8.50

JFETs

BFW11, BFW10, TIS73L, J177, J113, U1994, U1898, 2SJ176, J201	£1.00
J175, J176, J112, J111, 2N3820, 2N5467, BF244, 2N5460, J230	£0.60

Small power output/driver transistors

2SB649A, D669A, 2SA1208, 2SC2910, MJE253G, MJE243G, 2SA1725, 2SC4511	£1.20
BD139,BD140, BD135, BD156, BD435, BD436, MPSA63, BCV46	£0.50

MOSFETs

ZVP2106A	£0.30
Dual-gate 3SK45, BFS28	£1.50

Metal-cased transistors

BC143	£0.35
2N1711	£0.50
BCY71	£0.30
BC109C	£0.60

Amplifiers

LM318 high-speed op amp	£0.35
µA709 metal-cased op amp	£2.00
5534H metal-cased op amp	£3.00
LM384, TDA2030A, TDA2050V	£1.50
TAA435 (Mullard power amp driver for AD161/AD162)	£1.20

Synthesiser ICs

That/dbx 2180 VCA/VCF	£6.00
CA3080 VCA/VCF (vocoder)	£2.80
CA3280 VCA/VCF	£4.00
LM13600/ LM13700 VCA/VCF	£2.00
CA3086 transistor array	£1.00
PT2399 echo/delay	£1.50

Dual transistors

2N2639, 2N2223, 2N2910 (NPN)	£4.00
E401 (JFET Moog)	£4.00
2SK2145-Y dual JFET	£0.80
2N5564 JFET	£8.00
HN3C51F, HN3A51F	£1.00
DMMT3904/6, HN1A01F, HN1C01F	£0.50

Loudspeakers

PE Mini-Monitor

Volt PE165 6.5-inch woofer (each)	£85
Morel MDT29 tweeter (each)	£25
Kit *pair* of PE165/MDT29 plus Volt crossover parts and PCBs	£299
Monacor DT-28N tweeter (each)	£35
Vifa 19mm BC20SC15-04 tweeter (each)	£15

Volt crossover inductors

1.2mH, 1.5mH, 2mH, 2.7mH, 0.5mH (tapped at 0.3mH) (each)	£5.85

Fully assembled and tested high-quality speaker prototypes – ask for details

LS3/5As and other similar speaker systems (pair)	£200-£350

Fully tested reclaimed speakers

Vifa BC14 5-inch woofer (each)	£10
Vifa TC26 1-inch tweeter (each)	£10

Low-price speakers

Philips 4-inch 4070	£2.00
EMI 10x6-inch, 30Ω Alnico	£7
64mm 64Ω neodymium	£1.20

Capacitors

Note '10/63' denotes '10µF 63V'.

Polyester

3.3/100, 4.7/250, 4.7/63	£1.00
5.6/63, 8.2/63, 10/63	£2.00
Mullard 'Mustard' C296 0.22/400	£2.00

Polycarbonate

Axial 2.2/63 1%, 4.7/160, 6.8/63	£1.00

Radial

6.8/160V, 10/63	£2.00
Reclaimed 22/63	£2.00

Polystyrene

Philips 1% 4.7nF/160, 6.2nF/500, 12nF/63, 22nF/63, 110nF/63, 24nF, 2nF	£1.00
RIFA 1% 100nF/100, Suflex 90.9nF 0.5%	£2.00
Suflex 2.5% 10nF/63 (rad. or ax.)	£0.50

Radiation resistant

Siemens cellulose acetate MKL 2.2/25	£0.80

Electrolytic – Mullard blue 017 series

10/25, 22/25, 100/10	£0.50
150/40, 470/40, 1000/40	£1.00

Tantalum – axial metal cased

22/50, 47/35, 68/25, 100/20, 120/10, 150/16, 220/10, 330/6	£2.00
22/35, 33/35, 47/20, 68/15, 100/10, 150/6	£1.25
4.7/50, 6.8/35, 10/25, 10/35, 22/15	£1.00

Axial moulded-case tantalum

Kemet axial 6.8/10	£0.30
Kemet radial 33/10	£0.40
STC radial 100/20	£1.50

Tantalum bead

22/50	£1.00
470/3	£2.00
680/6.3	£3.00

Wet tantalum

220/25 axial	£3.20
Castanet button 140/30, 470/3	£3.20
Hughes 540/10	£3.20

Bipolar

Hermetic bipolar tantalum 16/35	£3.20
Elcap axial 10/50	£0.50
Generic radial 100/16, 470/35, 100/ 63, 22/35, 4.7/35, 220/16	£0.50

Philips solid-aluminium (axial) 121/123

47/16	£1.00
330/6.3	£2.00
100/35	£4.00

Philips Pearl 122 series (radial)

10/16	£0.30

Silvered mica (radial)

1nF/500 1%	£1.00

Trimmer capacitors

Vishay plastic-film 4-40pF	£0.80
Vishay plastic-film 5.5-45pF	£1.00
Vishay plastic-film 5-80pF	£1.20
Vishay plastic-film 10-250pF	£2.00
Mica 1-12pF, 2-40pF	£1.00

Audio transformers and inductors

Eagle transformers

LT44, LT722 driver, LT700, LT723 500Ω output	£2.50
LT30 500mW output	£3.50
5:1 interstage	£1.50

Repanco

T/T3 splitter transformer	£4.00
CH2 5mH RFC	£2.00

Balanced output transformer

Vigortronix 600Ω VTX-101-007	£10
Vigortronix 600Ω VTX-101-3001	£10
Vigortronix 600Ω VTX-101-3002	£15
Gardners 150Ω	£10
Reclaimed BBC LL74/MPC nickel core 600Ω	£12
Low-level 1:1 600Ω	£4
Reclaimed mic input transformer	£15

Inductors

82µH, 4.7mH, 100µH, 270µH, 10µH, 14µH (low Z)	£0.50
7-inch ferrite rod with MW and LW windings	£3.00

Special resistors

Bourns wire-wound trimmer 5kΩ 3059 JM panel-mount	£2.00
Thermistor RA53, R13	£4.00
A13	£2.00
Thermistor CZ1, CZ6	£1.50
Holco H2 2.2MΩ 1W, 1%	£1.00
Welwyn 1GΩ 2W	£1.00
5k Bourns 3321H cermet trimmer	£0.50

Potentiometers

Bourns 81

25kΩ lin cermet	£2.00
25kΩ lin, 5kΩ lin conductive plastic	£3.00

Bourns 91

10k dual-gang lin or log	£10.00

Plessey moulded-track

5kΩ log with switch	£3.00
50kΩ A/log	£3.00
Mil 250kΩ lin dual	£5.00

Alpha

16mm 4.7kΩ A/log	£0.80
220kΩ A/log	£0.80
10kΩ lin centre-detent dual-gang	£1.50

Allen Bradley J series/Honeywell

10kΩ lin 1 million cycles	£5.00
Blore Edwards AB 45 dual 5kΩ A/log with switch	£3.50
Alps RK9 dual-gang 5kΩ RD law	£4.00
Alps 50k log tapped motorised stereo	£5.00
BI P260 500kΩ log conductive plastic 1 million cycles	£2.00
Colvern wire-wound 100kΩ or 50kΩ dual-gang 3W	£5.00
Mil Spec hermetic 10Ω	£8.00

Miscellaneous

PCBs

Pocket Theremin (*EPE*, 1996)	£2.00
Elysian Theremin (*EPE*, 1996)	£6
Synth VCF, VCO (*EPE*, 2017)	£3.00
48V PSU (*EPE*, 2019)	£3.00
SMT dual transistor adapter	£0.40

Relays

STC hermetic 4190 24V DPDT	£4.00

Switches Blore Edwards/Plessey

MH1 4-pole 3-way	£3.00
Elma 01 2-pole 6-way	£6.00
3PDT guitar pedal stomp switch	£2.50

Valves

EC88, PCL84	£3.00

Contact

Jake Rothman
Manchester House
Market Street, Craven Arms
Shropshire SY7 9NN
(visit by appointment)

+44 (0)1597 829102

jrothman1962@gmail.com

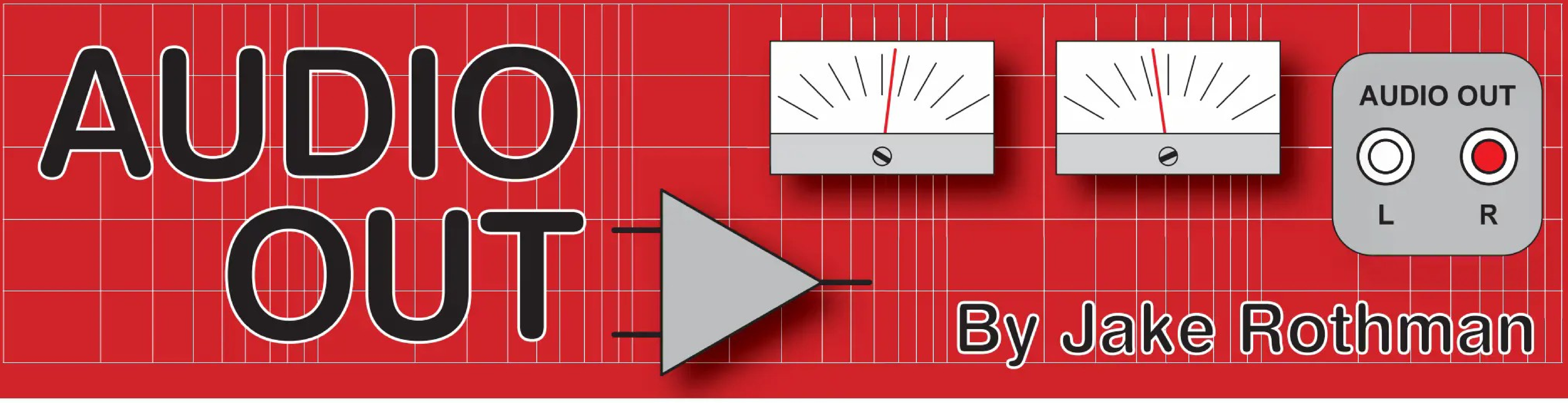
Minimum order £5.00 inc post
Quantity discounts negotiable

Payment

PayPal, bank transfer, cheques (payable to "J Rothman", UK pounds only)
No VAT payable

Postage

Small Jiffy bag £2.99
Small package £4.99
Big boxes and overseas at cost – ask for a quote



AUDIO OUT

By Jake Rothman

A complementary cascode amplifier

In the analog audio world, amplifier circuits mostly all look the same. That is because it's a mature technology and most engineers have settled on recycling circuits from Douglas Self's books, simply because they're proven to give excellent results at a reasonable cost.

Unfortunately, this gives little scope for creativity. I enjoy designing slightly obscure discrete circuits, especially for musical instruments, where they still come into their own.

For big Hi-Fi audio amplifiers, it's difficult to beat Self's circuits without a further increase in complexity, such as Bob Cordell's DH-220C Mosfet power amp. However, there is still a creative window available between application-note design, such as the TDA2050 chip, and Self's "blameless" architecture.

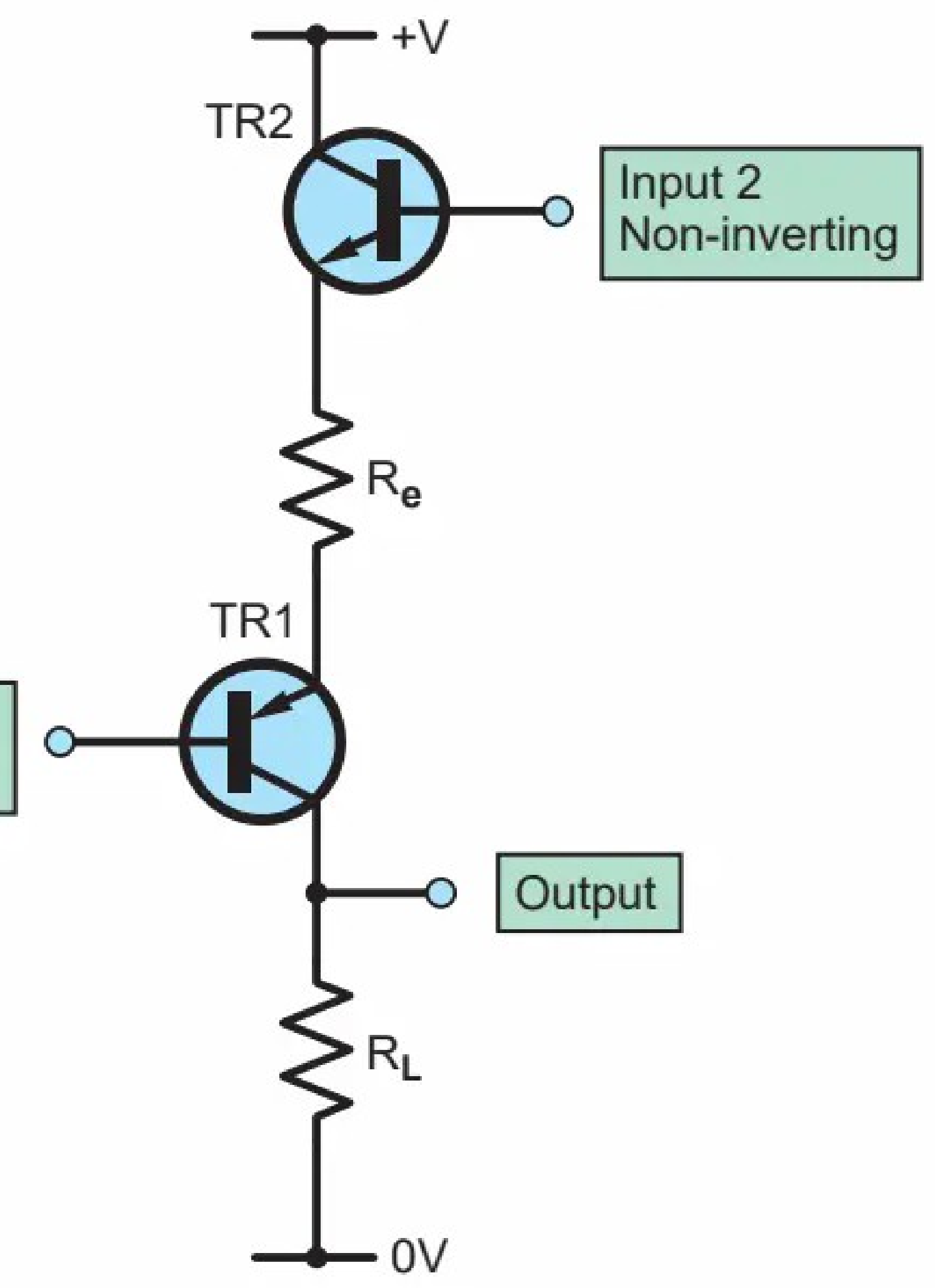


Fig.1: a complementary (NPN & PNP) cascode. It can be considered similar to a standard long-tailed pair.

This is where the complementary cascode circuit comes in (Fig.1), along with other early transistor circuits from the 1960s. It offers the AC performance of the long-tailed pair (LTP) with only two transistors rather than five, if a current mirror and sink is included.

This complementary cascode configuration could be considered a type of complementary LTP. The catch is it exhibits a DC offset of 1.2V, which is why it is rarely used. This is because the two base-emitter (V_{BE}) drops add up rather than cancel. Having said that, fixed offsets don't matter if the amplifier is configured as a single-rail AC-coupled design with an adjustable half-rail bias.

The complementary LTP needs no constant-current circuit since there is no tail current to consider, because one emitter directly feeds the other. Also, only one emitter resistor (R_e) is needed. With the values shown in Fig.3, the gain is reduced to around 2.2× from 100×, with a commensurate reduction in distortion and an increase in signal handling capability.

The even-order distortion cancellation is not as good as with the LTP, though, since the pair matching of complementary transistors is never going to be as good as same polarity devices. While low levels of even harmonic distortion are not a subjective problem in audio, it will likely make the total harmonic distortion (THD) higher on the specification sheet.

Complementary cascode buffer

Before designing a power amplifier using a new circuit topology, it makes sense to start with a simpler practical circuit, like a buffer. As with the LTP, most cascodes have two inputs, with one input often connected to a fixed bias voltage.

With the complementary cascode, one input is inverting and the other non-inverting. As in op amp circuits, the inverting input is often used for negative feedback. Note that in this circuit, the voltage amplifier stage (VAS), a common emitter stage, is inverting. So the inputs to the cascode have to be flipped.

To reduce the distortion of tantalum and electrolytic output coupling capacitors, negative feedback (NFB) can be used, as shown in the op amp circuit in Fig.2. The 22nF capacitor has to be a low-distortion film type. The circuit also needs to be loaded to avoid a very low-frequency hump in the response.

This technique can also be applied to the complementary cascode buffer shown in Fig.3. A similar technique is used for the power amplifier speaker coupling capacitor.

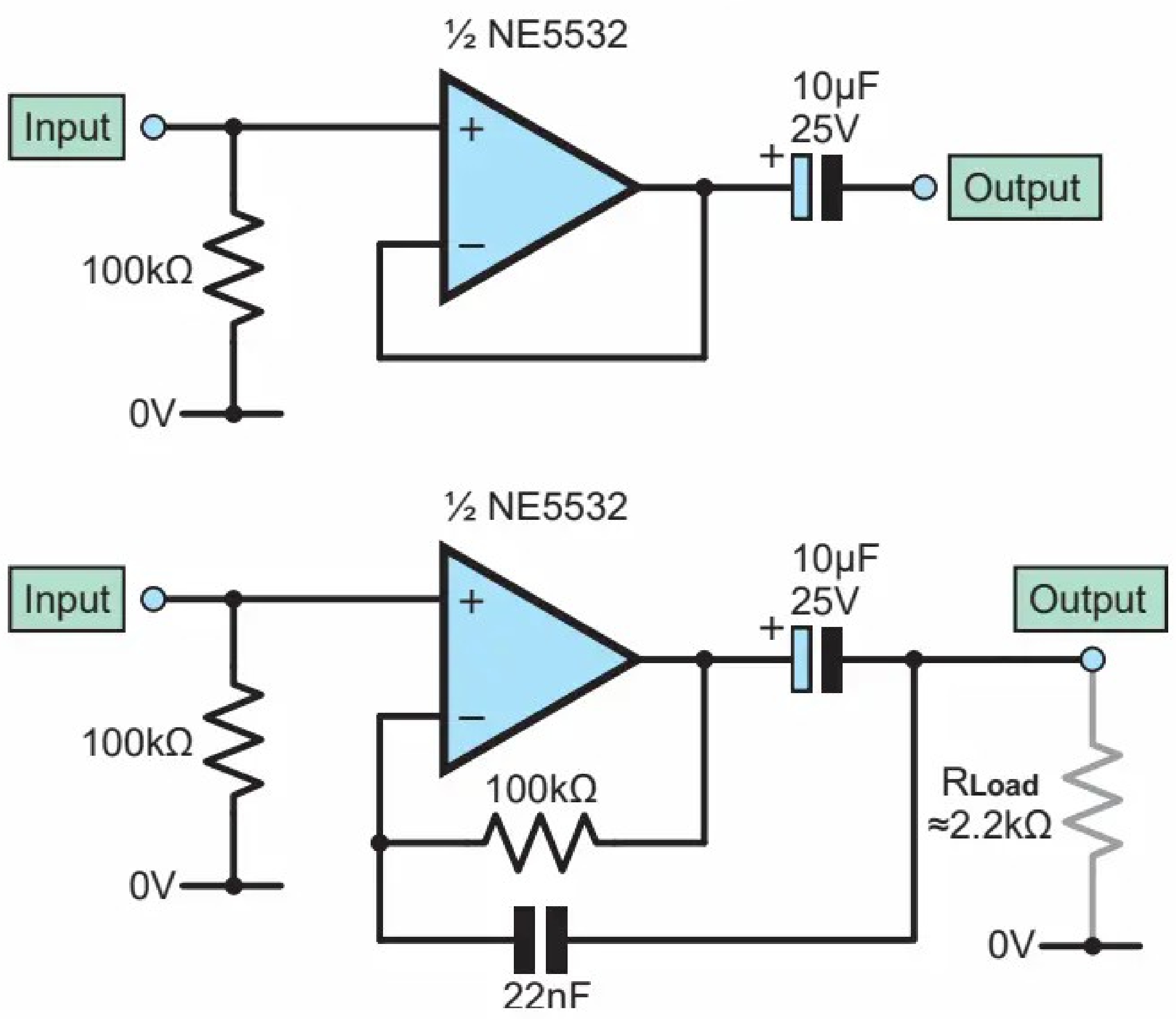


Fig.2: one of my favourite negative-feedback tricks I developed to avoid distortion from output electrolytic caps.

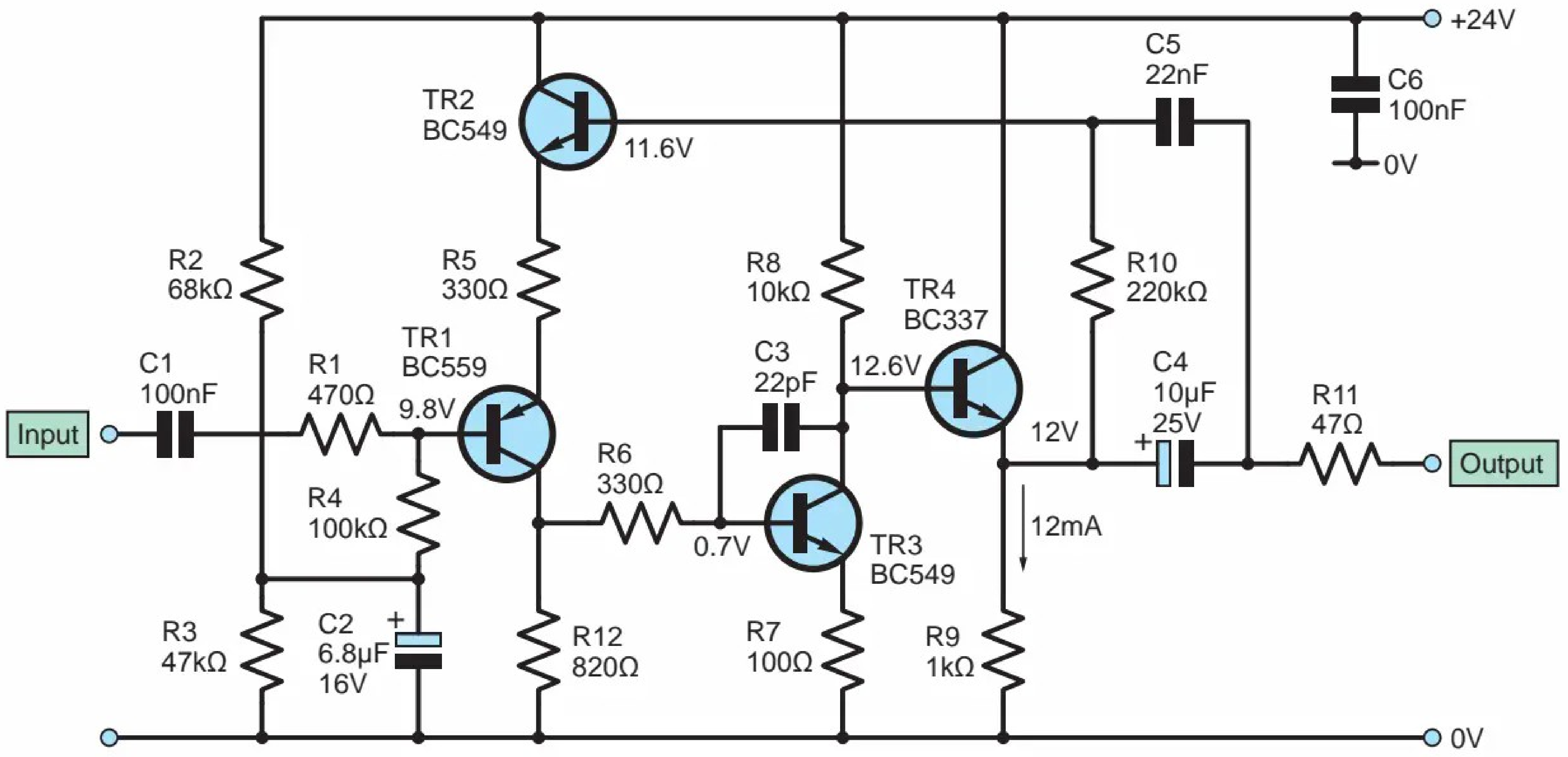


Fig.3: a buffer amplifier using a complementary cascode as a complementary long-tailed pair. I breadboarded this to prove the technique works.

If gain is required, the NFB can be reduced using a potential divider. The amount of gain for low distortion is limited in this circuit, because the open-loop gain is low compared to an op amp. However, the gain can be increased by a bootstrapped or constant-current collector load for resistor R9.

The power amplifier

The first commercial power amplifier using the complementary LTP I had was a Bi-Pre-Pak module; readers over 60 will remember them being advertised in a two-page spread in PE. The Sterling Sound SS125, designed by ex-Sinclair engineer Richard Torrens in 1977, inspired this circuit.

It intrigued me because it had a 0.02% THD rating rather than the typical 0.1% or more at the time. I used it as a midrange driver amplifier, along with Ben Duncan's Hi-Fi News active crossover in 1980, to make a very successful sound system. I was only 18 at the time, and we had great parties until my Dad got fed up.

Loads of ladies who wanted to be medics turned up, but they paired up with those men destined to become dentists. Sadly, they got paid much more for drilling holes in teeth than I did for drilling PCBs.

A single supply rail

The traditional LTP is perfect for dual-rail DC coupled-amplifiers, where the speaker is connected directly to the output. With single-rail amplifiers, a DC-blocking capacitor is used on the output so that the half-rail bias does not appear as DC across the speaker.

I've found these capacitors can cause problems when used with amplifiers above 40W RMS because they can generate significant distortion from the high current passing through them. With single-rail designs, the capacitor is polarised at half rail, so the distortion from polarity reversal does not occur.

The output capacitor also gives reliable DC fault protection, and the output transistors are also protected from output short circuits at low volumes. This is because the negative feedback loop at DC isn't broken. Of course, if the amplifier is playing full pelt when a short occurs, the capacitor offers no protection.

One major problem with single-rail power amplifiers is that the signal ground 0V can become contaminated with the half-wave rectified currents from the lower output transistor. This means that for the same circuit, the THD is always higher. Greater care has to be taken with the wiring layout.

Output inductor

When feeding long loudspeaker leads, as in typical domestic Hi-Fi setups, the capacitance can cause power amplifiers to oscillate. A damped inductor network (L1, R21) is often used to isolate this. If the leads are short, say in a guitar amp combo or active speaker, it can be omitted. The Zobel network alone, C12 and R25, will suffice to maintain stability.

Here's a funny aside. I knew one engineer who inadvertently relied on the parasitic inductance of the big output electrolytic (C10) to avoid using an inductor on the output. He got his comeuppance when the capacitors were "improved" with a lower equivalent series resistance (ESR) and the amps oscillated and blew up.

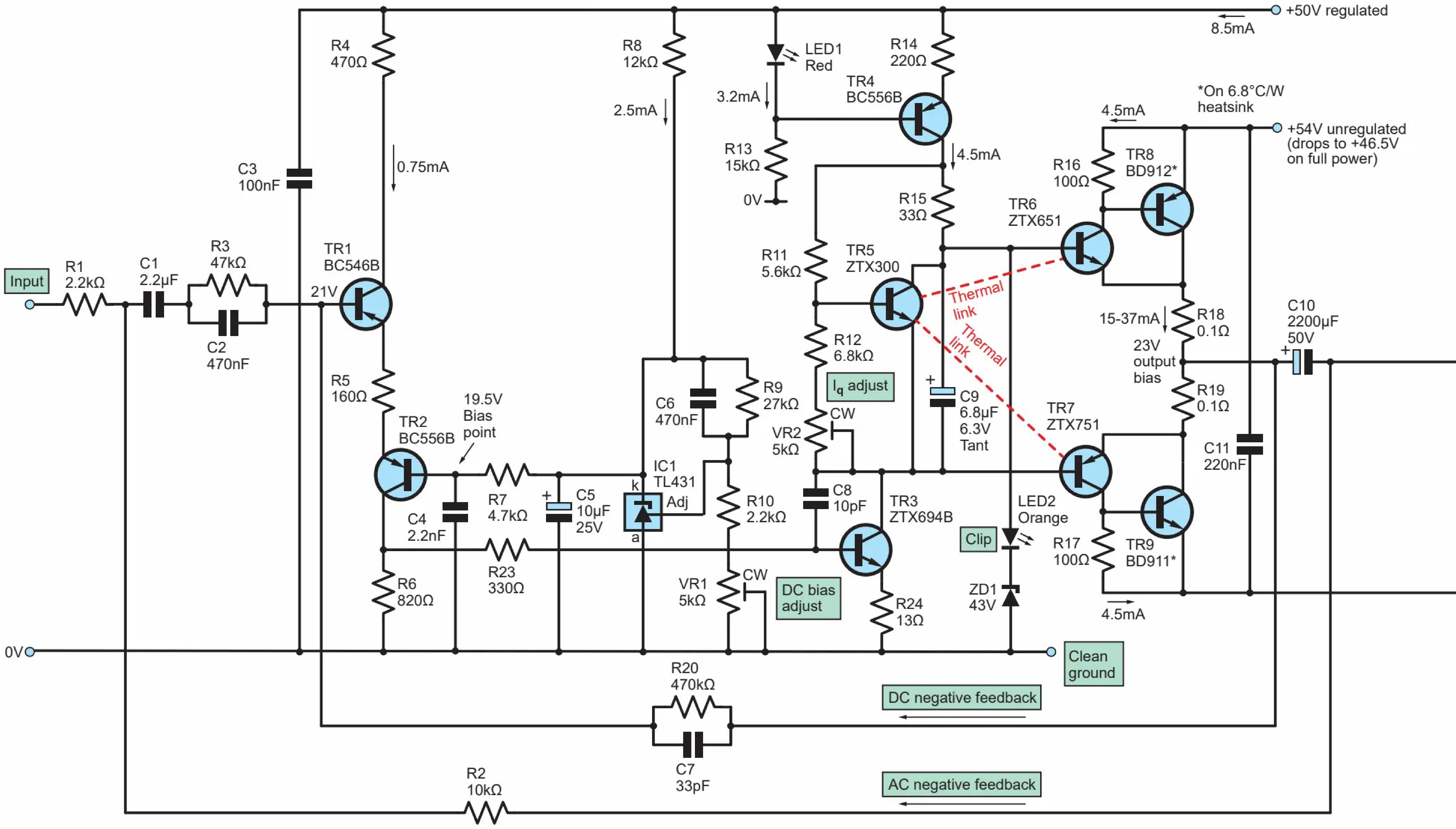
An interesting characteristic of the complementary cascode that I noticed is that it has less switch-on/off thumps compared to the LTP, which tends to 'snap on'. Power on/off transient generation is an area of analog circuitry that has not been fully researched yet (perhaps this would be a good topic for someone's PhD).

As a designer of active loudspeaker systems with up to eight power amplifiers, I will do anything to get rid of thumps, thereby avoiding the unreliability of speaker protection relays.

Bias generator

Let's now look at the features of the complementary cascode power amplifier circuit I've designed, shown in Fig.4.

For low-level circuits such as preamps, a resistor-based potential divider to generate the half-rail bias is fine. For a Hi-Fi class AB power amp running on a typical unregulated supply, that is not good enough. This is because the inevitable power supply voltage fluctuations can be fed back into the amplifier via the bias voltage, causing it to modulate itself.


+50V regulated
8.5mA
R4 470Ω
R8 12kΩ
LED1 Red
R14 220Ω
TR4 BC556B
3.2mA
2.5mA
R13 15kΩ
0V
4.5mA
*On 6.8°C/W heatsink
4.5mA
+54V unregulated (drops to +46.5V on full power)
R16 100Ω
TR8 BD912*
R15 33Ω
TR6 ZTX651
C3 100nF
0.75mA
R11 5.6kΩ
TR5 ZTX300
Thermal link
Thermal link
Input
R1 2.2kΩ
C1 2.2µF
R3 47kΩ
C2 470nF
TR1 BC546B
21V
R12 6.8kΩ
15-37mA
R18 0.1Ω
C10 2200µF 50V
23V output bias
R5 160Ω
19.5V Bias point
C6 470nF
R9 27kΩ
Iq adjust
CW
VR2 5kΩ
C9 6.8µF 6.3V Tant
R19 0.1Ω
TR2 BC556B
IC1 TL431
k
Adj
a
R7 4.7kΩ
C5 10µF 25V
C4 2.2nF
R10 2.2kΩ
C8 10pF
TR3 ZTX694B
LED2 Orange
TR7 ZTX751
C11 220nF
Clip
R17 100Ω
TR9 BD911*
R6 820Ω
R23 330Ω
VR1 5kΩ
CW
DC bias adjust
R24 13Ω
ZD1 43V
4.5mA
0V
Clean ground
R20 470kΩ
DC negative feedback
C7 33pF
R2 10kΩ
AC negative feedback

This raises distortion and causes the speaker cone to flap about at low frequencies, an effect I call 'bias pumping'. If you look at the data sheets for some power amp chips, it will give a higher distortion figure for single-rail use compared to dual-rail (unless it's been measured with a stabilised bench power supply). The solution is to regulate the bias voltage with a zener diode.

In this case, a TL431 'programmable zener' IC is used because it is adjustable (with VR1). If a regulated supply is used for the input stage, a decoupled resistive divider will suffice.

It's always a good idea to design an amplifier, its power supply and the speaker to be used as one system. I trim the bias for symmetrical clip at the lowest dip in the impedance curve of the speaker. Do it quickly to avoid burning out the driver.

Input and VAS power supply

This might be considered an over-the-top upgrade, but it gives an audible difference to me. As I'm now 62, I can no longer hear the extreme high-frequency distortion harmonics that most complex power amplifier circuits aim to minimise. Despite that, I still measure high-frequency distortion. I'm also very aware of hum and its modulation effects.

It's very difficult to regulate the power supply to the whole amplifier; it's often more complex than the amplifier itself, and it reduces the maximum output on peaks by 30%. Luckily, regulating the low-level input and voltage amplifier stage parts of the circuit is quite easy.

In this design, the output noise and ripple was reduced from 10mV to 2mV by adding regulation for the early, low-power stages. Use of a more complex LM317 high-voltage regulator circuit rather than the simple zener scheme (ZD4, ZD5) used here will reduce it still further.

If you want to add rail regulation to a dual-rail amplifier, two regulators will be needed, but with a single-rail amplifier, you need only one. The famous Quad 303 single-rail amplifier had full regulation, although it was mainly used to limit the maximum voltage on the output transistors. The complexity of regulation means it's rarely done.

(Dual regulated audio power amp supplies are available, such as those by Hypex, but being SMT switch-mode designs with over 100 parts, I've found them to last only last five years and be uneconomic to repair. That is unacceptable in the recording industry.)

Also, with current mirrors and other improvements to the input and driver circuitry, the common-mode rejection of the amplifier can be made very high. However, this only works until the output transistors saturate at clipping; then all the power supply noise is let through, giving very dirty-sounding clipping.

This horrid effect can be avoided by clean-clipping the driver stage first, but this will again reduce the maximum output power on peaks by around 30%. This is because the clipping level must be set below the lowest voltage that the unregulated power supply sags to on kick drums and other peaks.

As usual with electronics, there's a nice solution: since the current requirements are only around 10mA, a simple voltage doubler can be used with a zener regulator, and we can then set the rail voltage a few volts higher than the output stage supply rail.

A defined 'clean soft clip' level before the output stage can then be set by a zener diode (ZD1) with one LED in series to indicate clipping. The power is still reduced from 30W RMS into 8Ω to 26W, though.

Since low-power amplifiers, like this, will inevitably clip occasionally in real use, this modification is clearly audible and sounds excellent as a guitar amplifier. It also allows us to use most of the available voltage from a given power supply transformer.

With the transformer being the most expensive component in any power amplifier, it more than pays for itself. The Hi-Fi enthusiast's technique of using a 100W power amplifier to avoid dirty clips when driving small speakers seems a very expensive solution.

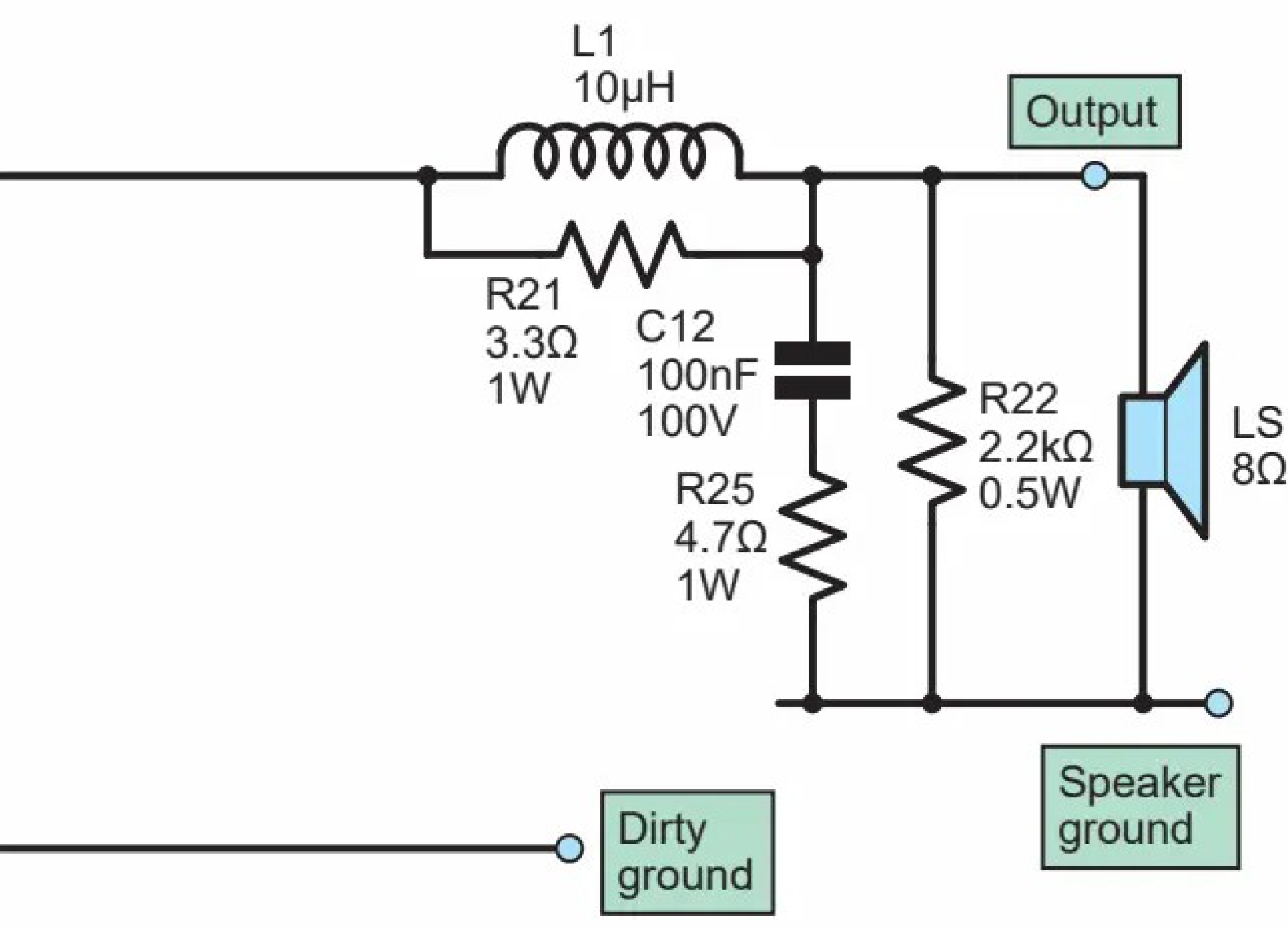
Faster switch-off

One of the problems with the traditional linear power supply setup is that it does not switch off quickly; it carries on working until the capacitors discharge. This can let through thumps from previous stages. By turning off the low-power regulated supply quickly, the power amp can be muted.

This is accomplished by a discharge circuit (TR1 & TR2) that pulls down the rail when the AC from the mains transformer goes away. This is driven by its own rectifier diode, D3, with minimal smoothing provided by capacitor C4, so it powers down before the rest of the circuit.

The power supply circuit is shown in Fig.5.

An 18-0-18V 50-60VA transformer is more than sufficient for one channel. A top-notch approach to prevent Earth loops in a stereo amplifier, would be to use a 35V + 35V 100VA dual secondary



◀ Fig.4: the complete power amplifier circuit. The currents through the driver transistors are about 4.5mA because the output transistors are not fully switched on under quiescent conditions. R3 and C2 form an LF step network to prevent the LF hump.

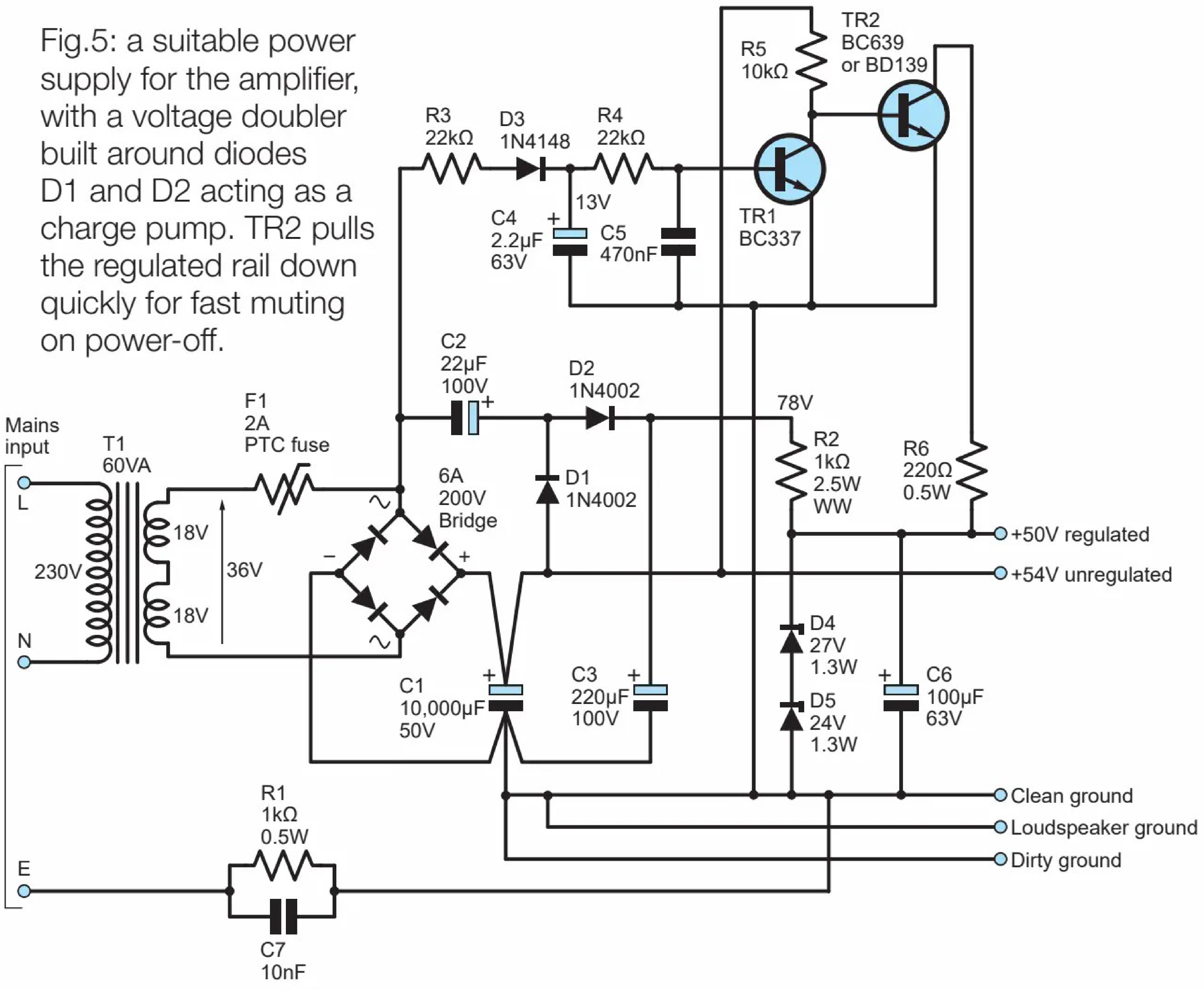


Fig.5: a suitable power supply for the amplifier, with a voltage doubler built around diodes D1 and D2 acting as a charge pump. TR2 pulls the regulated rail down quickly for fast muting on power-off.

toroidal transformer driving two separate PSU boards.

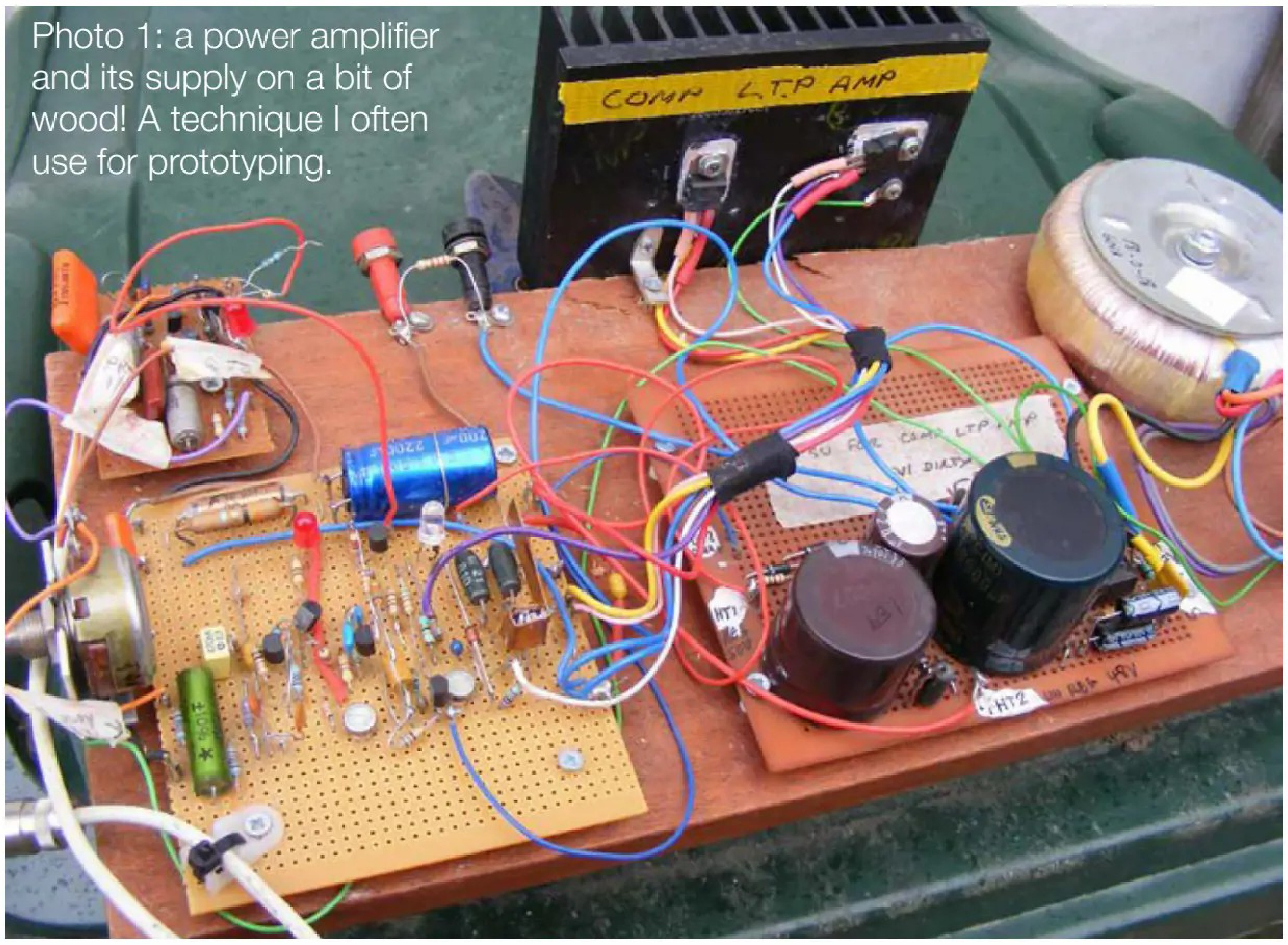


Photo 1: a power amplifier and its supply on a bit of wood! A technique I often use for prototyping.

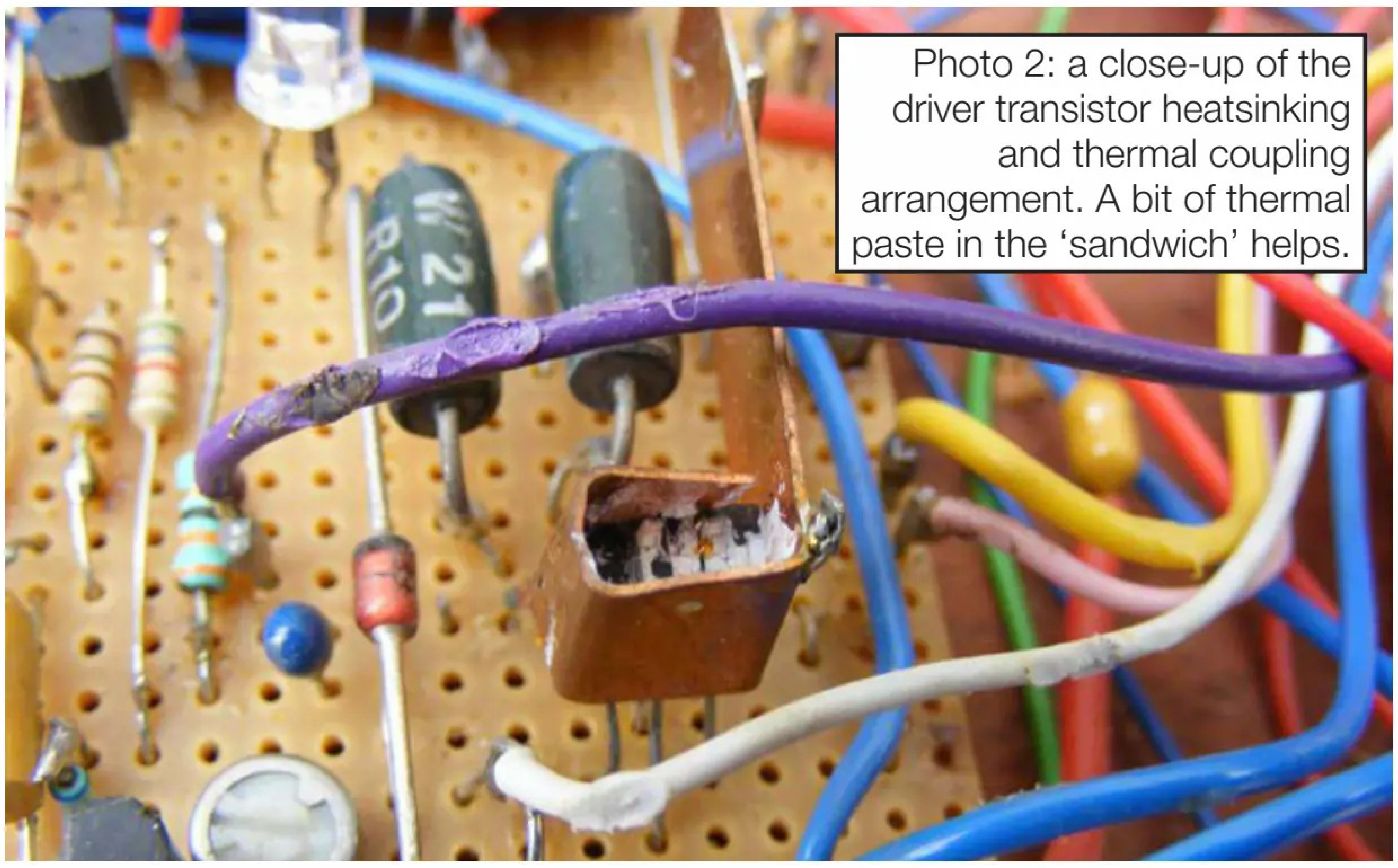


Photo 2: a close-up of the driver transistor heatsinking and thermal coupling arrangement. A bit of thermal paste in the 'sandwich' helps.

Voltage amplification stage (VAS)

This is just the standard configuration, a common-emitter stage with a dynamic load. The load here is a constant-current source, rather than a bootstrapped collector resistor, because it gives lower crossover distortion. Its higher voltage loss is obviated by use of the higher voltage rail from the voltage-doubled supply.

One problem with the standard common-emitter VAS is that it is subjected to the full rail voltage swing between emitter and collector (V_{CE}). With bad examples of some older types of transistors, such as the BFY51 and BC107, this could result in distortion from the Early effect. This is where the gain and capacitance of the transistor is modulated by its V_{CE} voltage.

I remember having to buy special Motorola transistors that avoided this problem. These were used in Bailey and Quad amplifiers. Another source of good audio transistors was Ferranti in Oldham, near Manchester. Their ZTX E-line series 'Zetex' transistors were used in the groundbreaking PE Gemini and Orion amplifiers. They are still in production by Diodes Incorporated 50 years later!

2A 60V ZTX651 and ZTX751 transistors are used in the driver stage here. The Orion VAS transistor was a Ferranti BFS61 or ZTX451. I have found the ultra-cheap BC546B only works okay for swings up to 30V. For this amplifier, I used a ZTX694B with a 13Ω emitter resistor, which exhibits little Early effect. It has a V_{CE} rating of 120V.

High-voltage transistors always have less Early effect, but often the H_{fe} is low; the ZTX694B has an H_{fe} of 400. Adding a small amount of emitter resistance (R24) also helps reduce distortion at high levels. Conversely, adding resistance increases distortion at low levels due to open-loop gain reduction.

I did try a cascoded VAS stage based on the Gemini circuit, but I found it offered no improvement on the ZTX694B.

D. S. Gibbs and I. M. Shaw, who designed both PE amplifiers, were Ferranti application engineers at the time. Shaw was also the inventor of a diode dodge to improve the symmetry of quasi-complementary output stages. Later, this was improved by Baxandall, then adopted commercially by Rogers and Naim.

Low gain for low noise

Most power amplifiers have too much voltage gain for use in active speakers; typically 20× to 40×. This boosts the noise from the active filters, creating a noticeable hiss. Peter Baxandall reduced the gain of the Quad 405 amplifiers while increasing the crossover level in the pioneering KEF KM1 active speaker, resulting in a system as quiet as a passive speaker.

So with this amplifier, I also decided to go for a much lower gain. I've even made one unity-gain stable so you can use it as an active filter itself, or put an active gain control around it.

I also made it inverting, which allows a fully filtered half-rail bias generator to feed the non-inverting input (base of TR2). Inverting amplifiers generally have more noise than their non-inverting counterparts, but this is only if you need a high input impedance.

In this case, the input impedance is kept low, at 2.2kΩ; this is high enough for most op amps and reasonable discrete circuits to drive. The inverting configuration is also useful for bridging with another power amplifier and for adapting to a balanced input. Now we have something sufficiently different from other amplifier circuits to make the design work worthwhile.

Output stage

I've opted to use the complementary emitter follower pair (CFP) output configuration here since it works well for powers below 50W, where the output transistors can be fairly small, with low capacitance. For powers above this, and where devices need to be paralleled, I use the Darlington emitter follower (EF) topology, which gives lower distortion and has better HF stability.

By Darlington, I don't mean monolithic Darlington transistors, where there is a driver and output transistor on the same chip, since this exacerbates thermal instability. The driver transistors have to be separate from the output transistors to avoid being heated up by them. Philips used monolithic Darlingtons successfully in their active speakers by putting a thermistor in the V_{BE} multiplier.

For small amplifiers, the CFP gives

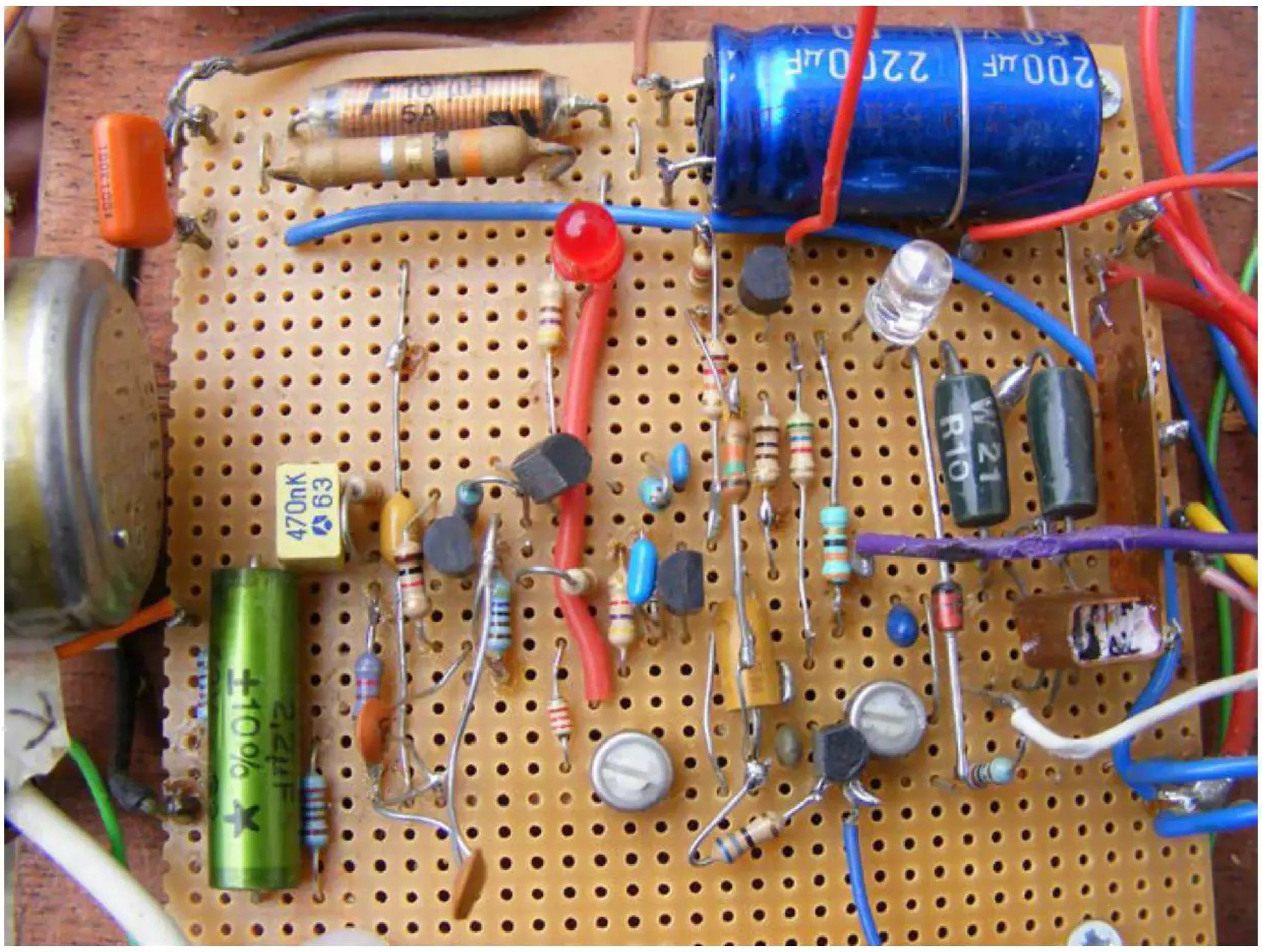


Photo 3: the prototype stripboard is slightly messy because the board is too short.

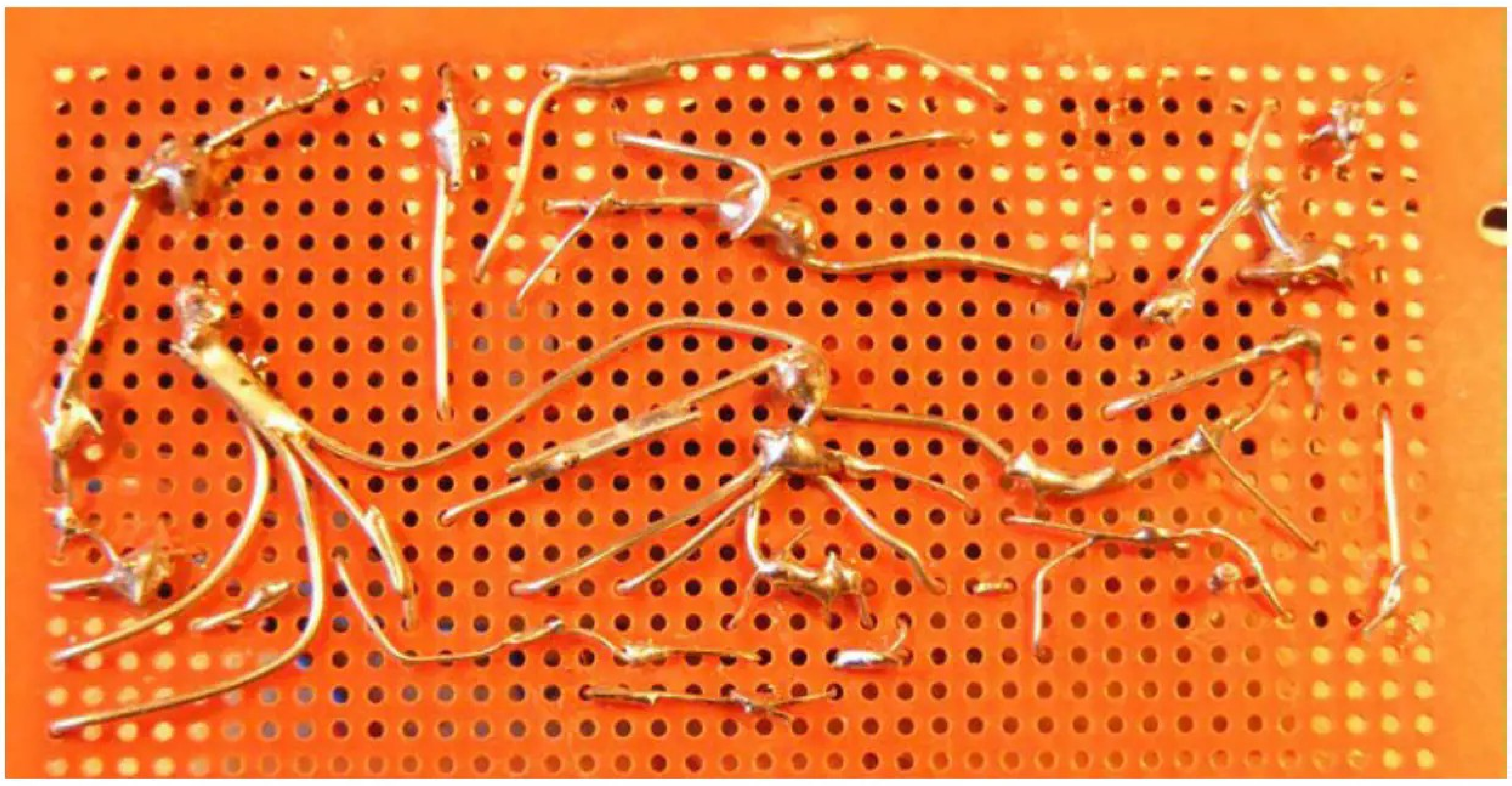


Photo 4: the star wiring to capacitor pins is visible on the power supply underside.

lower quiescent dissipation. Also, the V_{BE} multiplier transistor only needs to be in thermal contact with the drivers for thermal stability, avoiding the hassle of mounting it on the main heatsink.

The crossover distortion of the CFP is more 'spiky' than the Darlington EF and occupies a smaller voltage swing. It also has poorer stored charge removal, hence the HF distortion is a bit higher at 10kHz. The collector load resistors on the drivers are quite important for low distortion at high levels. Generally, for amps of this size, 100Ω is optimum, setting the driver current to 4.5mA.

The output transistors TR8 and TR9 are not critical. I've used old TIP41/2C, BD911/912 and even the venerable TIP3055/2955 complementary pairs. Of course, if you want lower HF distortion, you could use modern, faster types.

Building and testing

The complete prototype assembly is shown in Photo 1. Yes, it is a real "breadboard"; a nice metal enclosure is not warranted for an initial prototype (the mains wiring has to be well-insulated!).

I always build prototype power amplifiers on stripboard; breadboards are just too prone to bad connections and short circuits, which could cause it to blow up. Normally, stripboard is only suitable for simple circuits, which is true if one is going to build it in one go.

The key to building a complex, fragile circuit on stripboard is to build it in testable sections. As always, electronics can be easily partitioned into its constituent stages, and this is the key to its success as an engineering discipline. Try doing this when building a railway bridge, though, and life becomes much more difficult!

Always use a bigger board than you need. If you run out of holes, you have to start soldering onto component leads, which quickly becomes messy. The board can always be snipped down to size with side-cutters (or cut with a hacksaw) later.

Although a good PCB layout designer does not usually place components in the same physical position as the schematic, with strip board prototypes, it makes it easier to test if it's wired like a circuit diagram. That is with the positive rail is at the top, the negative rail on the bottom and the signal flow from left to right. The stripboard is shown in Photo 3.

If you have to move something to optimise the layout, always test it before adding something else. Once you have more than two errors, the time taken to get things working rises disproportionately.

Add loops of tinned copper wire to attach test probes and crocodile clips. Veropins don't work; the spring-loaded test leads can ping off. An extra-big loop should be used for the Earth, since there may be up to five leads attached.

For this amp and many others, I've built and tested the circuit blocks in the following order. Always ramp up the voltage slowly using a bench power supply and set it to a low current limit, around 250mA:

1. Create the current source bias voltage using LED1. This tells you circuit is powered.
2. Add the current source, TR4. Make sure the current is correct; in this case, 4.5mA.
3. Add the input transistors, TR1, TR2 and VAS TR3.
4. Test the gain with feedback resistors R1 & R2 wired into the circuit.
5. Add V_{BE} multiplier transistor TR5 and test that the VR2 preset works. Set the voltage across its collector and emitter to a minimum.
6. Add driver transistors TR6 and TR7. Since this is basically a push-pull emitter-follower, the feedback can be connected and a basic test performed.
7. Finally, add output transistors TR8 and TR9. Gradually increase the current limit on the supply.

To thermally couple TR5, TR6 and TR7, sandwich V_{BE} multiplier TR5 between the two drivers (TR6 and TR7), wrapping them all together with a copper strip (see Photo 2). This also acts as a low-power heatsink. I used another Zetex transistor (a cheap low-spec one) for TR5 because the flat E-line shape allows them to fit together with maximum surface coupling.

Adjust VR1 for symmetrical clipping with a 50W 8Ω load resistor and VR2 for the lowest distortion at low powers (3V peak-to-peak). This will typically give a quiescent current of 15-37mA.

The great thing about getting a stripboard circuit working is that you can guarantee the next developmental stage, a proper PCB, will work better. It will have lower parasitic capacitance and resistance.

Power supply construction

Stripboard is less than ideal for building power supplies since it is very difficult to make star connections on capacitor pins as shown for C1 in Fig.5 This technique is necessary to reduce the hum induced by charging pulses and prevent noisy Earth currents from impinging on the audio. Power supplies also need low resistance tracks.

It is possible to buy 0.2-inch (5.08mm) Veroboard, which is usable, but it's rare. It's best to use perfboard (which has no copper) with tinned copper wire. This technique can give better results than a PCB, probably because the tinned copper wire has a substantially larger cross-section than a PCB track with standard thickness. I occasionally resort to soldering tinned copper wire onto PCB tracks to lower their resistance. The underside is shown in Photo 4.

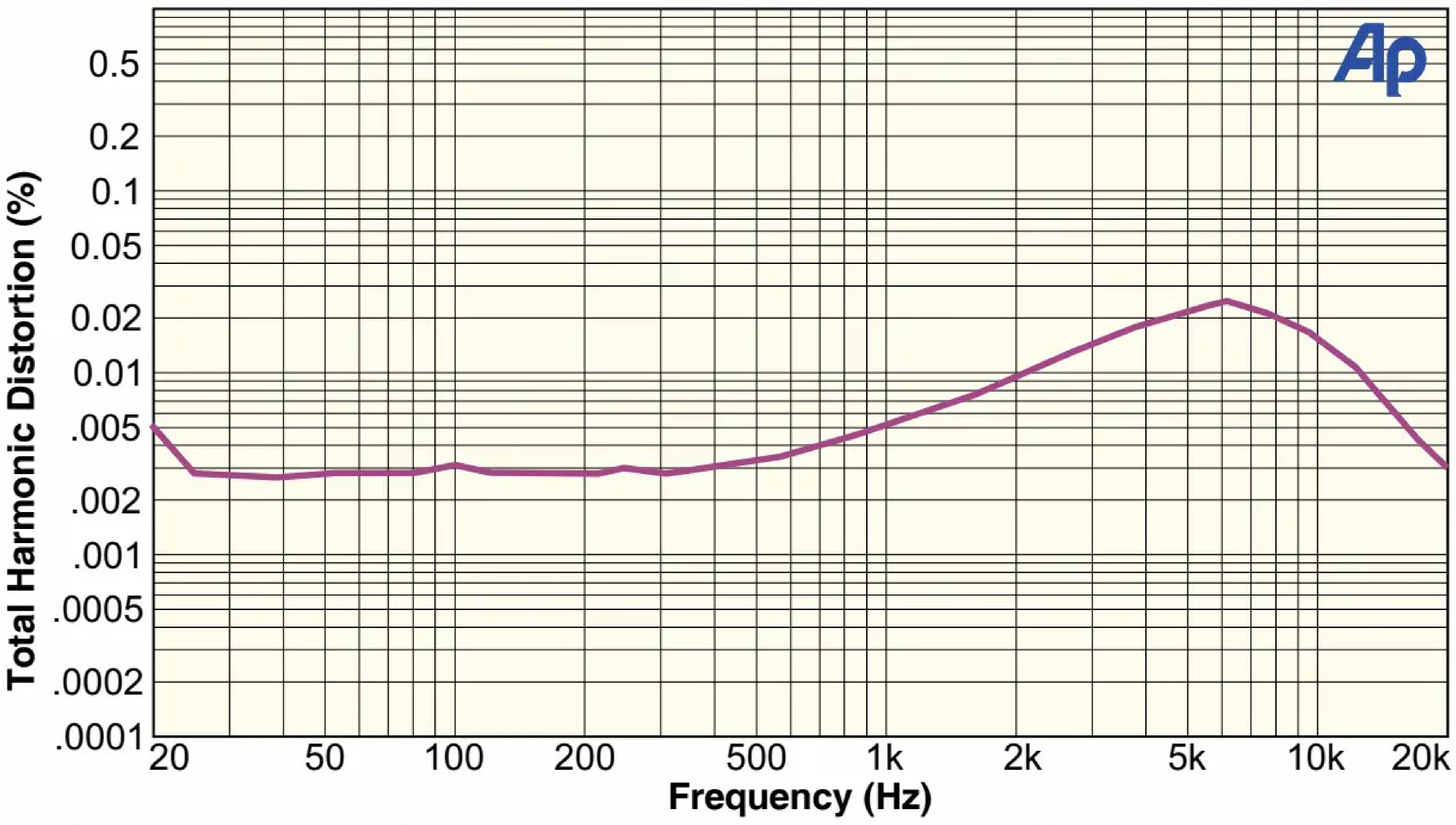


Fig.6: the distortion vs frequency curve for the power amplifier. This was done at low power (2W into 8Ω).

Performance

The distortion vs frequency curve of an amplifier is always revealing; this one is shown in Fig.6. The THD+N is around 0.003% midband at low powers, so the complementary LTP works almost as well as a traditional LTP. If a single input transistor stage had been used, the distortion would have been about 0.01%.

There is little increase in distortion at 20Hz, which illustrates the effectiveness of NFB around the output capacitor. There is a rise to 0.02% at 10kHz at all levels, which is not so good. This suggests the amplifier is running out of open-loop gain at high frequencies. This is because the compensation, layout and output stage need further optimisation.

The complementary LTP's subtraction for the negative feedback also possibly doesn't work so well at higher levels compared to a properly matched LTP. The LF/mid-band distortion of the amplifier steadily increased to 0.005% at 12.5W and to 0.018% at 20W, so we were back to the stated specification of the Stirling Audio module. However, there could be some other problems, such as in the component layout.

I've effectively reinvented the wheel here, and it does not give a Douglas Self level of performance. However, it's still worth adding the complementary LTP to the armoury of discrete building blocks, along with this small power amplifier system suitable for active speakers.

These days, linear power amp chips are often single-sourced and expensive compared to a handful of discrete transistors, so if you have the time, roll your own amp! **PE**

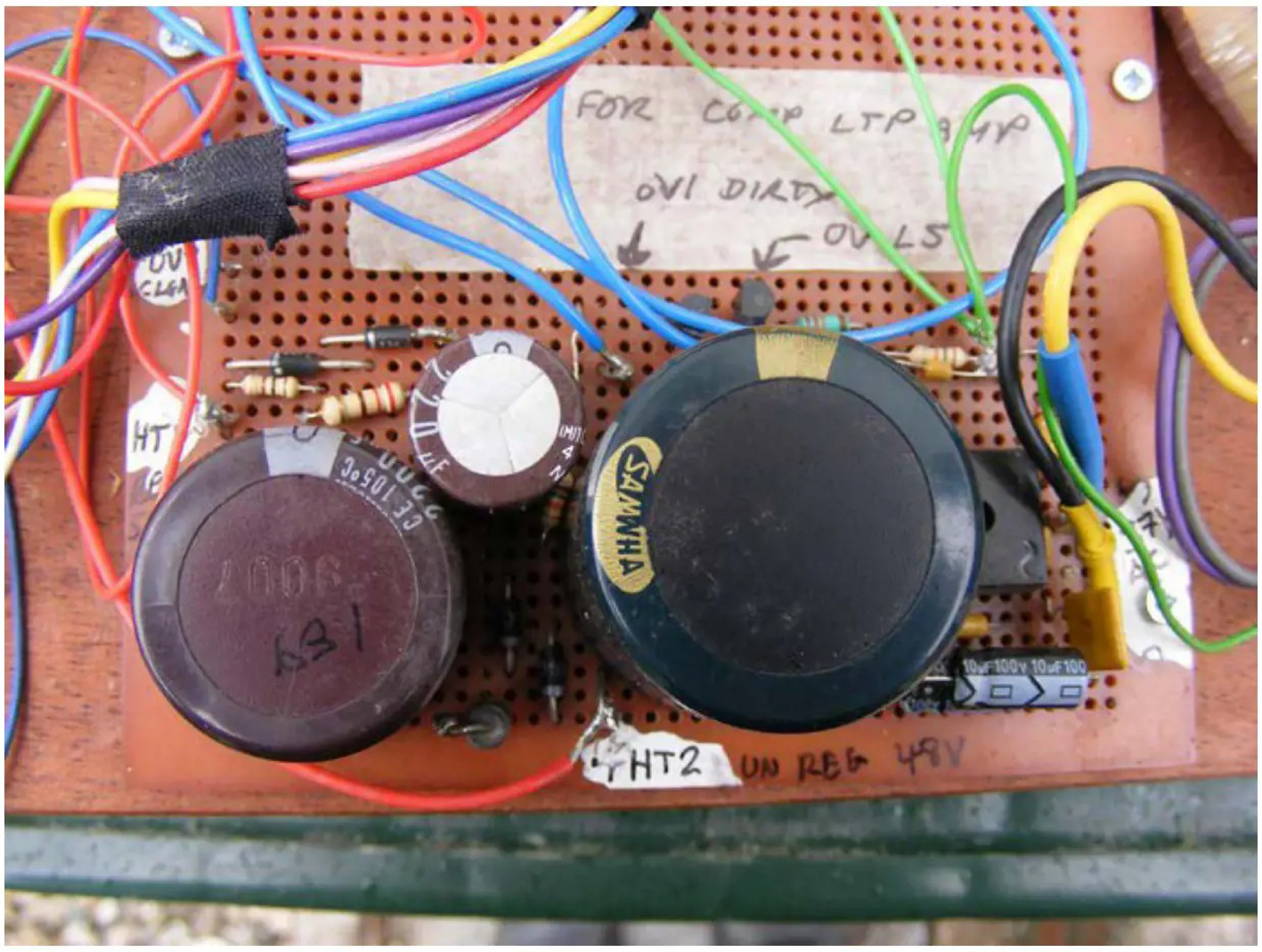


Photo 5: the power supply. Note the connections going directly to the pins of the big reservoir capacitor, C1.

HAMMOND®

1591 ABS flame-retardant enclosures

Learn more:
hammondmfg.com/1591

uksales@hammfg.com • 01256 812812

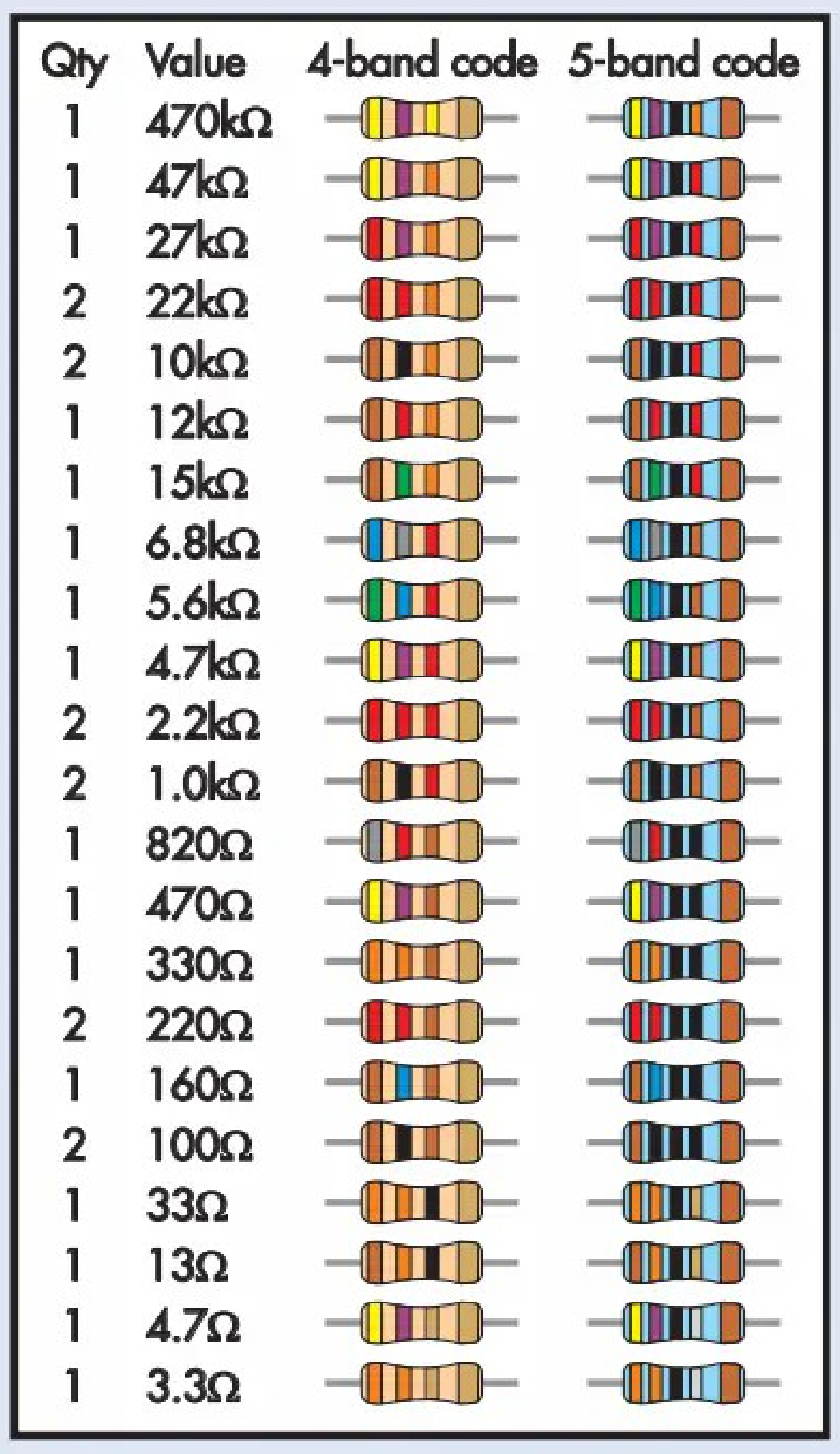
Parts List – Complementary Cascode Amp

1 6 × 5in (150 × 125mm) piece of stripboard
2 5kΩ trimpots (VR1, VR2)
1 10μH 5A suppression inductor (L1)
2 6.8°C/W flag heatsinks
1 10mm length of 0.5mm-thick copper strip (from builder's merchants)
various screws, nuts etc

Semiconductors
1 TL431 programmable voltage reference IC, TO-92 (IC1)
1 BC546B 65V 100mA NPN transistor, TO-92 (TR1)
2 BC556B 65V 100mA PNP transistors, TO-92 (TR2, TR4)
1 ZTX694B 120V 1A NPN transistor, E-Line (TR3)
1 ZTX300/ZTX108 45V 100mA NPN transistor, E-Line (TR5)
1 ZTX651 60V 2A NPN transistor, E-Line (TR6)
1 ZTX751 60V 2A PNP transistor, E-Line (TR7)
1 BD912 100V 15V PNP transistor, TO-220 (TR8)
1 BD911 100V 15V NPN transistor, TO-220 (TR9)
1 3/5mm red LED (LED1)
1 3/5mm orange/amber LED (LED2)
1 43V 400mW zener diode (ZD1) [eg, BZY88C43V43V]

Capacitors (all ±20% tolerance unless noted)
1 2200μF 50V low-ESR electrolytic (C10)
1 10μF 25V electrolytic (Al or Ta) (C5)
1 6.8μF 6.3V tantalum (C9)
1 2.2μF 63V polyester/MKT (C1)
1 470nF 63V polyester/MKT (C2)
1 470nF 50V X7R ceramic (C6)
1 220nF 50V X7R ceramic (C11)
1 100nF 100V polyester/MKT (C12)
1 100nF 50V X7R ceramic (C3)
1 2.2nF 50V X7R ceramic (C4)
1 33pF ±5% NP0/C0G ceramic (C7)
1 10pF ±5% NP0/C0G ceramic (C8)

Resistors (all ¼W ±5% or better)
1 470kΩ ±1% metal film
1 47kΩ
1 27kΩ
1 15kΩ
1 12kΩ
1 10kΩ
1 6.8kΩ
1 5.6kΩ
1 4.7kΩ
1 2.2kΩ ½W
2 2.2kΩ
1 820Ω
1 470Ω
1 330Ω
1 220Ω
1 160Ω
2 100Ω
1 33Ω
1 13Ω
1 4.7Ω 1W
1 3.3Ω 1W
2 0.1Ω 1-2.5W (wirewound)


4-band code
5-band code

Qty	Value
1	470kΩ
1	47kΩ
1	27kΩ
2	22kΩ
2	10kΩ
1	12kΩ
1	15kΩ
1	6.8kΩ
1	5.6kΩ
1	4.7kΩ
2	2.2kΩ
2	1.0kΩ
1	820Ω
1	470Ω
1	330Ω
2	220Ω
1	160Ω
2	100Ω
1	33Ω
1	13Ω
1	4.7Ω
1	3.3Ω

Power supply parts
1 4in × 2in (100 × 50mm) piece of perfboard
1 18-0-18V 50VA toroidal or 37V 1.6A EI-core transformer
1 2A 60V PTC thermistor ('solid state fuse')

Semiconductors
1 BC337 45V 800mA NPN transistor, TO-92 (TR1)
1 BD139 80V 1.5A NPN transistor, TO-225AA (TR2)
1 200V 6A bridge rectifier (BR1)
2 1N4004 400V 1A power diodes (D1, D2)
1 1N4148 75V 200mA signal diode (D3)
1 27V* 1.3W zener diode (ZD4)
1 24V* 1.3W zener diode (ZD5)

* a single 51V zener could be used if available

Capacitors
1 10,000μF 50V solder-tag electrolytic (C1)
1 220μF 100V electrolytic (C3)
1 100μF 63V electrolytic (C6)
1 22μF 100V electrolytic (C2)
1 2.2μF 63V electrolytic (C4)
1 470nF 63V polyester/MKT (C5)
1 10nF 50V X7R ceramic (C7)

Resistors
2 22kΩ ±5% ¼W
1 10kΩ ±5% ¼W
1 1kΩ ±5% 2.5W wirewound
1 1kΩ ±5% ½W
1 220Ω ±5% ½W

NEW!

5-year collections

2019-2023

All 60 issues from Jan 2019 to Dec 2023 for just £49.95

PDF files ready for immediate download

2018-2022

All 60 issues from Jan 2018 to Dec 2022 for just £49.95

PDF files ready for immediate download

2017-2021

All 60 issues from Jan 2017 to Dec 2021 for just £49.95

PDF files ready for immediate download

2016-2020

All 60 issues from Jan 2016 to Dec 2020 for just £44.95

PDF files ready for immediate download

See page 33 for further details and other great back-issue offers.

Purchase and download at:
www.electronpublishing.com

Intelligent Dual Hybrid Power Supply

ON
OFF
U1,U2
U1,U1
Track
Power
V
I
VA
ADC
Vos
Scale
Cal

PART 2: BY PHIL PROSSER

Our new Dual Hybrid Supply has very quiet outputs given its use of switchmode regulators to provide good efficiency and high power output in a small package. The outputs can be used independently or together in series or parallel, all controlled through a single easy-to-use digital interface. We described the circuitry last month, so this article will concentrate on assembling and calibrating the Supply.

There are a few steps in assembling, testing and calibrating this Supply. First, you need to build the four PCB assemblies: two regulator modules, the control board and the front panel board. Then you need to wire them up and put them through some basic checks to make sure they are all functional.

Following that, you attach the regulator modules to the main heatsink, prepare the case, mount everything in the case and wire it all up. Once you've done that, we'll take you through the calibration procedure, which is mainly done via menus on the LCD screen, with the aid of a decent multimeter.

There's quite a bit to get on with, so let's start with populating the regulator PCBs.

Building the regulator module(s)

Each regulator module is built on a double-sided PCB coded 18107211, measuring 116 x 133mm. Fig.10 is the PCB overlay diagram; it shows which components go where and indicates the correct orientations for polarised components. Refer to it as you build the board assembly.

If you are making a dual power supply, only one board needs the LM2575-5 (REG3) and associated components (L3, D12 etc) to be loaded. It is essential to only fit links LK1 & LK2 on the one board with the LM2575-5 regulator. So install those on one board now – you can use 0Ω resistors or lengths of tinned copper wire (Bell wire would work too).

Construction of the Regulator Module commences with all resistors, except for the 0.01Ω current sense resistor and 0.05Ω current sharing resistors. Leaving these off for now will mean that the board lays flatter on the bench, making it easier for you to solder the fiddly components that come later.

With those smaller resistors in place, mount all the diodes bar the TO-220 case diode and bridge rectifier, checking carefully that each is in the correct orientation before soldering.

Follow these with the 100nF film capacitors, 10µF electrolytic capacitor and remaining MKT capacitors. Before you fit the ceramic capacitors, solder the SMD ICs in place. Then install the small transistors (TO-92 package). Make sure you don't get the two different types mixed up.

We have described how to do this on many occasions. The basic idea is to tack one pin down, check that the placement and orientation are correct, add flux paste to all the pins, solder all the pins, then clean up any bridges which have formed using more flux paste and some solder wick.

Pay attention to the orientation of the MAX14930 isolators, IC6 & IC7; they are mounted in opposing directions. We have added markings near pin 1 of each SMD IC to assist.

With the SMD chips in place, fit the ceramic bypass capacitors.

Dual Hybrid Bench Supply, part two

There are two 15µF surface-mount tantalum capacitors on the top of the board.

These go with the positive end toward the regulator. Double-check their orientation; the positive end should have a stripe. There are also five surface-mount capacitors on the back side of the board; fit them next.

Now is a good time to load the components we held back: the 1W, 0.01Ω and 0.05Ω resistors. Then mount the headers, connectors and fuse clips and install the fuse.

Making the diode heatsink

TO-220 diode D3 needs a small heatsink to make it bulletproof. Its dissipation is only high if the Supply's output is short-circuited, but ideally we want it to handle that continuously.

We used a 55mm by 40mm piece of 1.6mm-thick aluminium with a fold in it. We recommend you do the same, as there is no need for a 'bought one', and this is the optimum size for the available space. Fig.11 shows how to fold and mount this heatsink.

Now fit the larger transistors (Q3 & Q10), three DIP ICs, plus the LM317, LM337 and LM2575-5 regulators. The regulators can be mounted with a couple of millimetres lead length. The +12V regulator (REG1) and negative regulator (REG4) need small flag heatsinks which are attached with an M3 machine screw, crinkle and flat washer, insulating bush and washer, as shown in Fig.12.

Before you mount the electrolytic capacitors, attach the heatsink to the TO-220 diode. This will make it easier

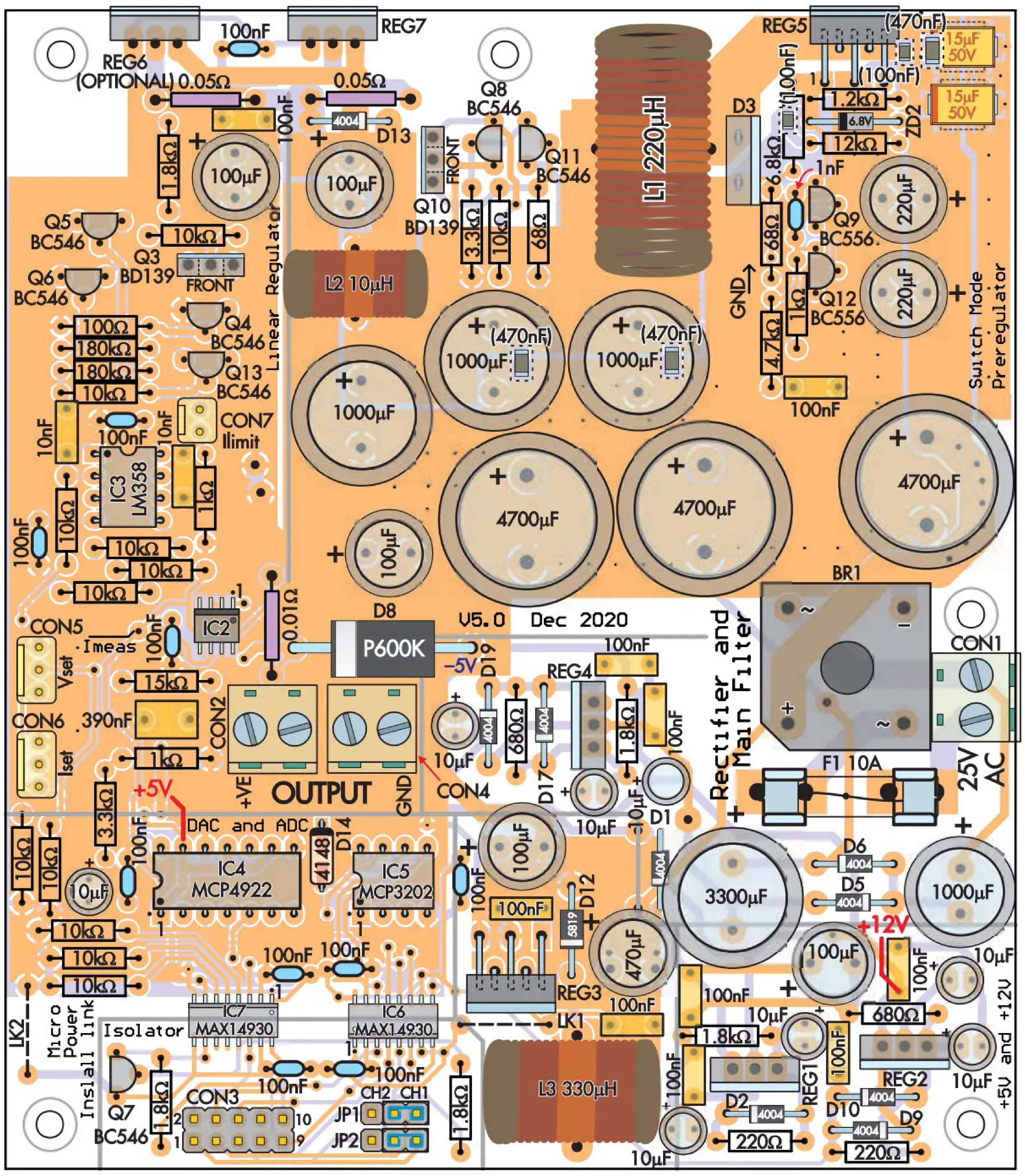
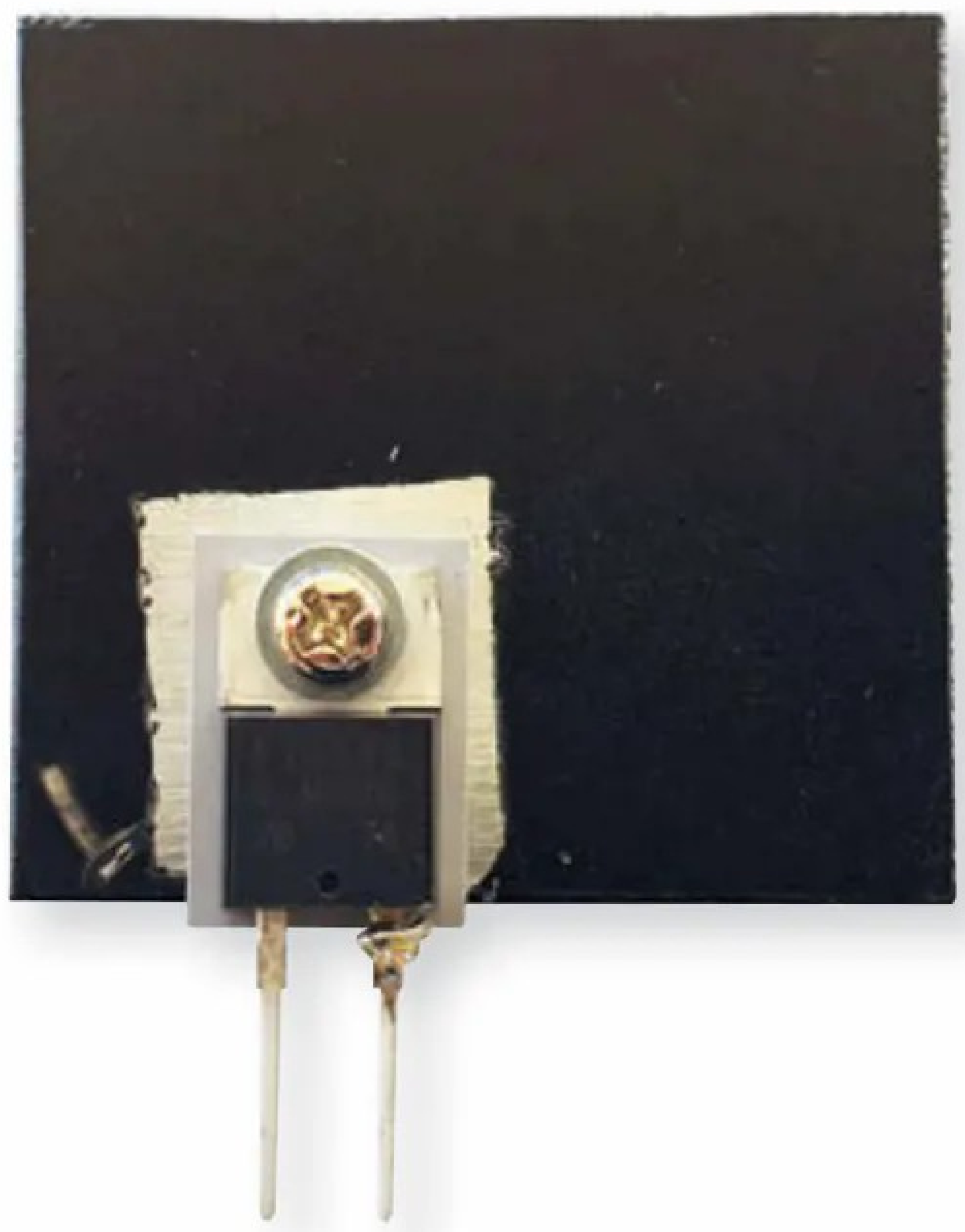

REG6
(OPTIONAL)
0.05Ω
100nF
REG7
0.05Ω
D13
Q8
BC546
Q11
BC546
Q10
BD139
FRONT
L1 220µH
REG5
(470nF)
15µF 50V
(100nF)
1.2kΩ
12kΩ
6.8V
ZD2
D3
6.8kΩ
1nF
Q9
BC556
Q12
BC556
220µF
GND
4.7kΩ
68Ω
1kΩ
Switch Mode Preregulator
Q5
BC546
Q6
BC546
Q3
BD139
Linear Regulator
L2 10µH
100µF
1.8kΩ
10kΩ
3.3kΩ
100Ω
180kΩ
Q4
BC546
Q13
BC546
CON7
Ilimit
10nF
IC3
LM358
1000µF
(470nF)
4700µF
BR1
CON1
F1 10A
25V AC
Rectifier and Main Filter
CON5
Imeas
Vset
IC2
0.01Ω
D8
P600K
V5.0
Dec 2020
-5V
D19
REG4
100nF
CON6
Iset
390nF
15kΩ
1kΩ
CON2
+VE
OUTPUT
GND
CON4
680Ω
D17
D1
+5V
DAC and ADC
IC4
MCP4922
D14
4148
IC5
MCP3202
D12
5819
3300µF
D6
D5
+12V
470µF
REG3
LK1
100nF
10µF
1.8kΩ
680Ω
REG2
+5V and +12V
LK2
Micro Power
Install link
Isolator
IC7
MAX14930
IC6
MAX14930
Q7
BC546
CON3
CH2
CH1
JP1
JP2
L3 330µH
D2
REG1
D10
D9
220Ω

Fig.10: the Regulator board is somewhat packed but not difficult to assemble. There are just a few SMDs; none with particularly fine-pitch leads. The only components that mount on the underside of the board are five SMD capacitors, all in the upper right-hand section. They are shown in an 'x-ray' fashion here.



Here is an example of how to mount the diode to the heatsink. Take note that the heatsink should have a bend in it as shown in Fig.11.

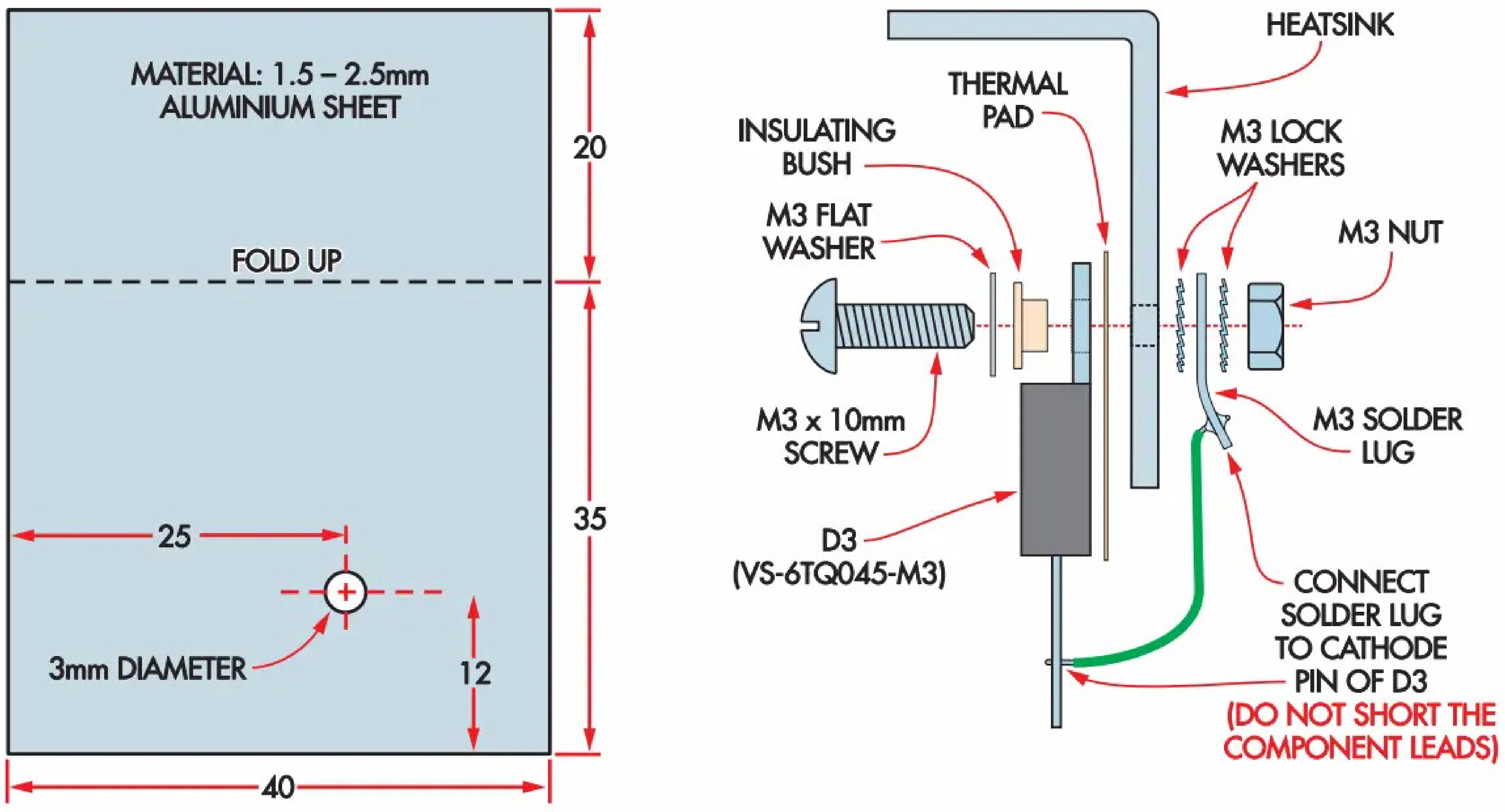
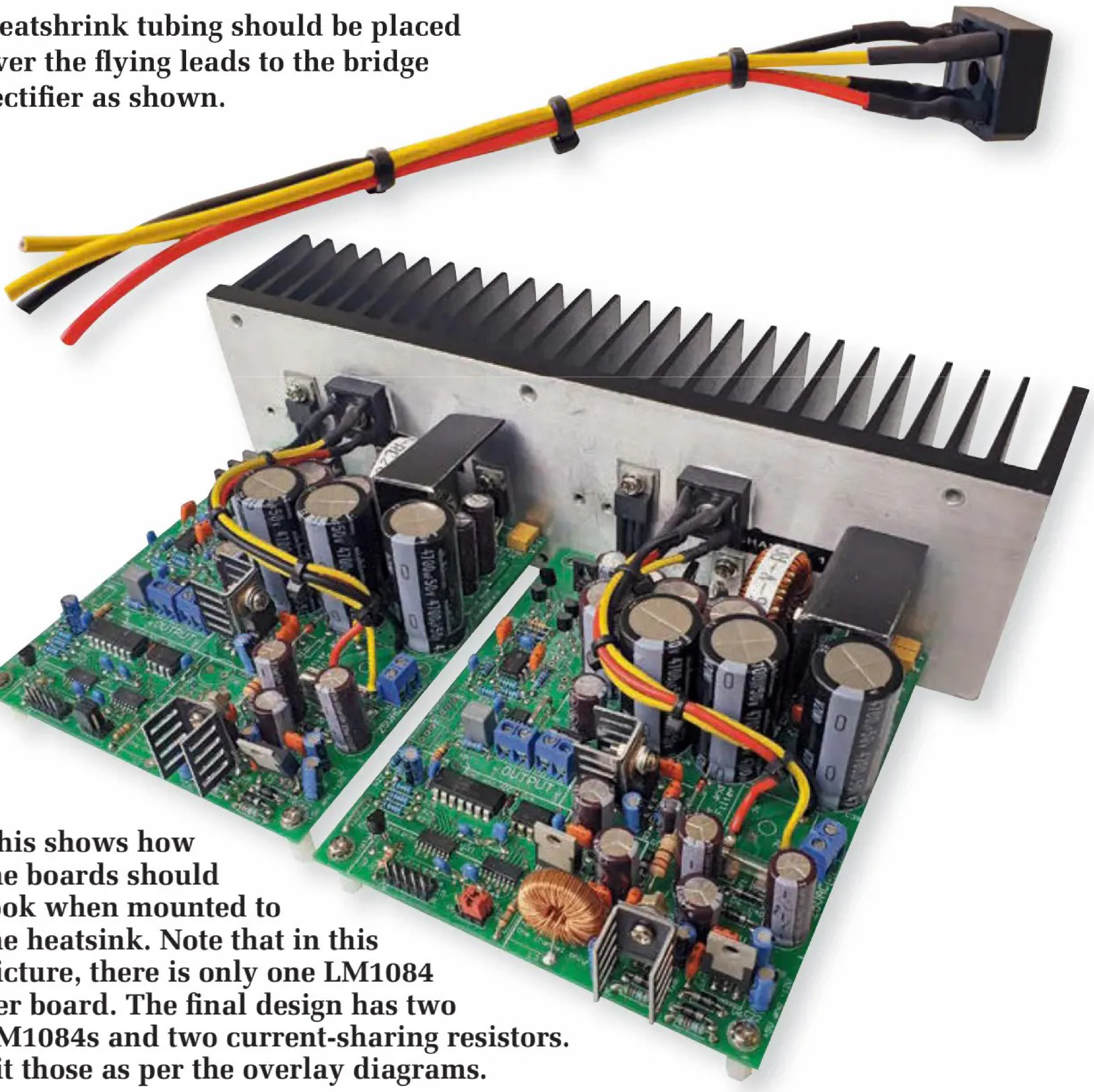

MATERIAL: 1.5 – 2.5mm ALUMINIUM SHEET
FOLD UP
20
35
25
12
40
3mm DIAMETER
HEATSINK
THERMAL PAD
INSULATING BUSH
M3 FLAT WASHER
M3 LOCK WASHERS
M3 NUT
M3 x 10mm SCREW
M3 SOLDER LUG
D3 (VS-6TQ045-M3)
CONNECT SOLDER LUG TO CATHODE PIN OF D3
(DO NOT SHORT THE COMPONENT LEADS)

Fig.11: we couldn't easily find a commercial heatsink to fit in the space around diode D3, so we made one. It's simple as you just need to cut out a rectangle of aluminium, drill one hole and fold it 90° where shown. Then attach it to the tab, including connecting the heatsink to the device's cathode for EMI reduction.



Heatshrink tubing should be placed over the flying leads to the bridge rectifier as shown.

This shows how the boards should look when mounted to the heatsink. Note that in this picture, there is only one LM1084 per board. The final design has two LM1084s and two current-sharing resistors. Fit those as per the overlay diagrams.

to get all the bits aligned and tightened. If you forget, you can poke a screwdriver through the hole in the toroidal inductor, but that is much more fiddly.

You can now fit the remaining electrolytics; put the larger ones in last as they tend to dominate the board. All bar two of these have the positive (+) lead toward the main heatsink, or to the left with the heatsink at the top. The two 220µF electrolytics do not follow this rule. These are at the input to the MC34167 and have their (longer) positive leads to the left, as demanded by the pinout of this device.

Finally, load the inductors. We put a dab of neutral cure silicone under ours to stop them moving, and recommend that you do the same. At this point, everything except the parts that mount to the main heatsink should be on the board.

The bridge rectifier is attached via 150mm flying heads, allowing it to be mounted to the heatsink. Put short lengths of heatshrink tubing over the connection of the flying leads to the bridge rectifier, as shown in our photo above.

Route the leads to the rectifier pads on the PCB. We used 15cm lengths of 7.5A rated hookup wire; red for positive, black for negative and yellow for AC. Also use heatshrink tubing to insulate the connections to the bridge rectifier leads.

Building the control boards

The controller for this project is the same one that was used for the Audio DSP project in the January-March 2020 issues and the Low-distortion DDS Signal Generator from February 2021. The main difference is that here, the PIC32MZ is programmed with the Intelligent Power Supply firmware.

The PCB overlay for this controller board is shown in Fig.13. We've removed most of the components you don't need for this project from that diagram, although it won't hurt if you fit them anyway.

As usual, fit the SMD ICs first (watch their orientation and check for bridged pins!), followed by the other SMDs, then the lower-profile through-hole components, finishing off with the taller parts.

Besides the ICs, be careful that the cathode stripes of the diodes go in the right locations, plus the SMD LED cathode (which is often marked with a green dot or T-shape). Also make sure that the positive (longer) leads of the electros go to the pads marked with + symbols.

If you will be programming your own microcontroller, the HEX file is available for download from our website. It can be programmed in-circuit via CON23, but note that if you plan to plug a PICkit in, it goes to the row of pins closest to the micro, with pin 1 at the end marked with a "1". Or you can purchase a pre-programmed micro from the Silicon Chip Online Shop, in which case you can skip that step.

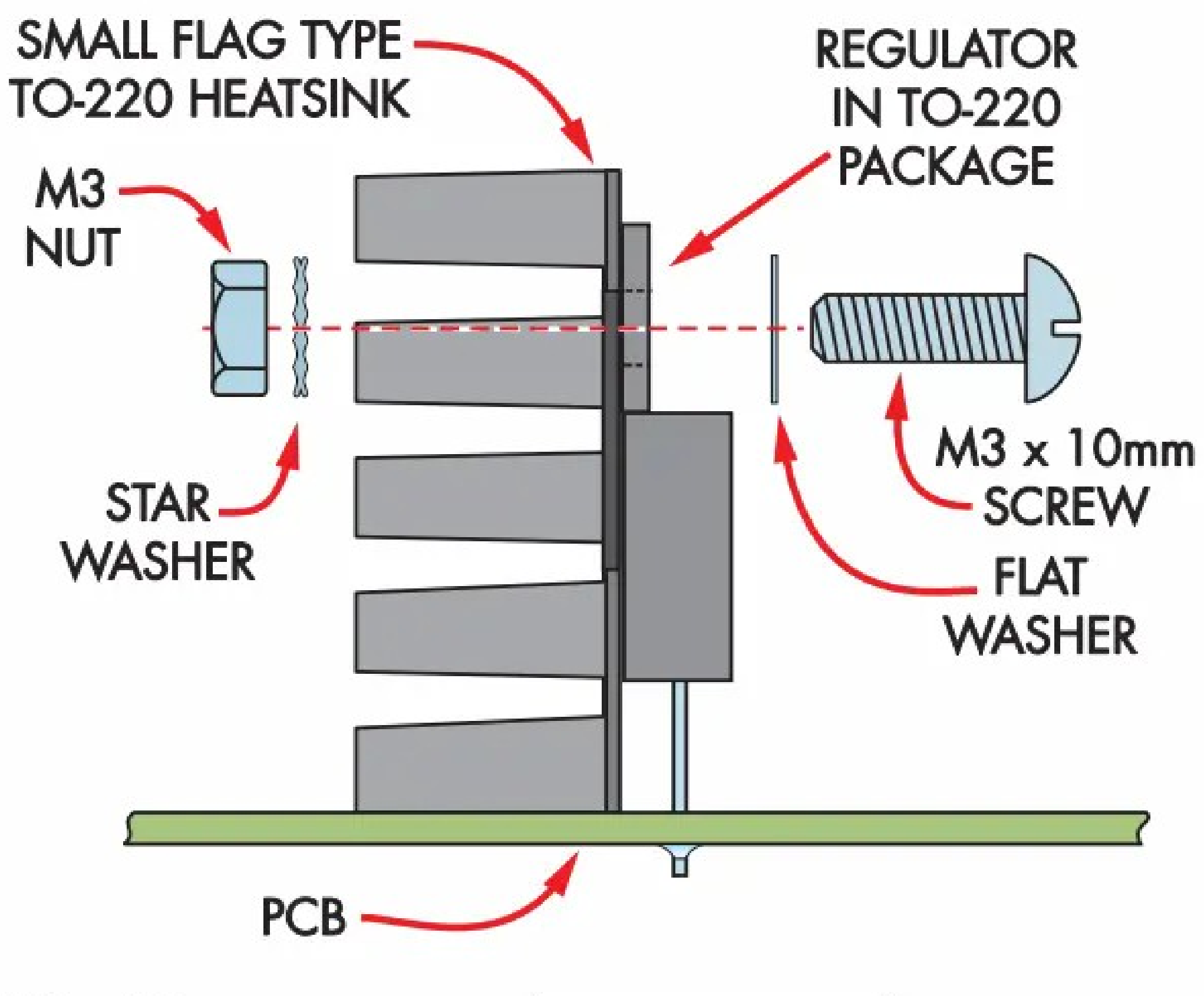

SMALL FLAG TYPE TO-220 HEATSINK
REGULATOR IN TO-220 PACKAGE
M3 NUT
STAR WASHER
M3 x 10mm SCREW
FLAT WASHER
PCB

Fig.12: we are using pre-made heatsinks for REG1 & REG4; attach them like this.

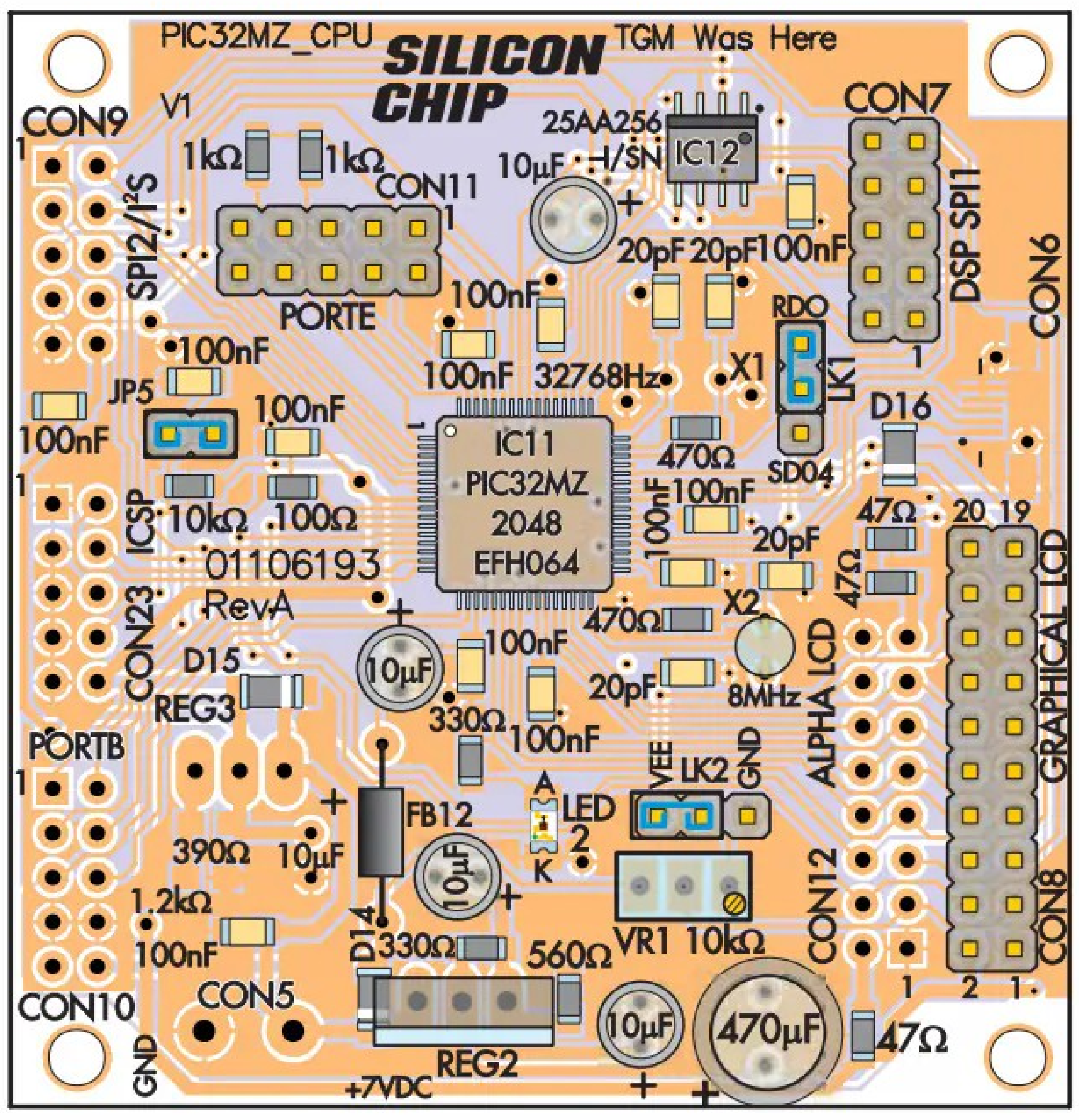

PIC32MZ_CPU
SILICON CHIP
TGM Was Here
CON9
V1
1kΩ
1kΩ
25AA256
H/SN
IC12
CON7
CON11
10µF
SPI2/I²S
100nF
20pF
20pF
100nF
DSP SPI1
CON6
PORTE
100nF
RDO
100nF
100nF
32768Hz
X1
LK1
JP5
100nF
100nF
D16
IC11
PIC32MZ
2048
EFH064
470Ω
100nF
SD04
ICSP
10kΩ
100Ω
01106193
RevA
20pF
47Ω
20
19
CON23
D15
470Ω
X2
47Ω
10µF
100nF
ALPHA LCD
GRAPHICAL LCD
REG3
330Ω
20pF
8MHz
PORTB
100nF
VEE
LK2
GND
FB12
LED
A
K
390Ω
10µF
CON12
1.2kΩ
10µF
100nF
D14
330Ω
560Ω
VR1 10kΩ
CON8
CON10
CON5
GND
REG2
+7VDC
10µF
470µF
47Ω

Fig.13: the powerful PIC32MZ-based control board for this project has been used in several previous projects. Some of the components are not needed for this one, so we have left them off this overlay diagram. Solder IC11 first (watch its orientation!), then IC12, followed by the passive SMDs (resistors & capacitors), then the remaining SMDs and finally the through-hole components.

Dual Hybrid Bench Supply, part two

This control module connects to both regulator modules with a multi-drop 10-way ribbon cable, which we'll make up shortly. It also connects to the front panel PCB, shown in Fig.14.

There isn't much to assembling this board. Just fit the two resistors, then the seven caps, followed by the header on the top. That just leaves rotary encoders and buttons, which mount on opposite sides. The encoders are on the top side and the pushbuttons on the underside. Make sure all of those are square and pushed down firmly before soldering their pins.

When it comes to the rotary encoder, any of the horizontally-mounting "TT" 20-pulse-per-revolution parts with a switch should work (Mouser part code 858-EN11-VSM1AF20 has been verified as working). These are available with either a D-shaft or spline shaft.

Once you have assembled this board, it's a good time to make up the three ribbon cables, as shown in Fig.15.

Cut the 10-way cable to 320mm and 250mm lengths and the 20-way cable to one 160mm length. Crimp on the IDC plugs as shown in the diagram. Note how the cable folds through the strain relief clamps at either end, but not on the sole middle plug. Some IDC plugs might not come with relief clamps.

These lengths assume you are using the recommended case and will be sticking to our layout. If you are varying either, you might need longer cables, so check that first.

Power supply assembly

Now it's time to fit this all into a neat benchtop case.

One of the design goals for this project was to keep interfaces and wiring simple and tidy. This is achieved by the PIC32 communicating with the regulator modules using an SPI interface. If you are into Arduino or Micromite, you could design your own controller.

The majority of work now is in preparing the case and heatsink. The case we have specified is an ideal size for the workbench, and provides a professional looking finish to the product.

You could use any other case of suitable size, with the only provisos being to ensure the case has adequate mechanical rigidity to secure the transformer and heatsink, and that it can be safely Earthed.

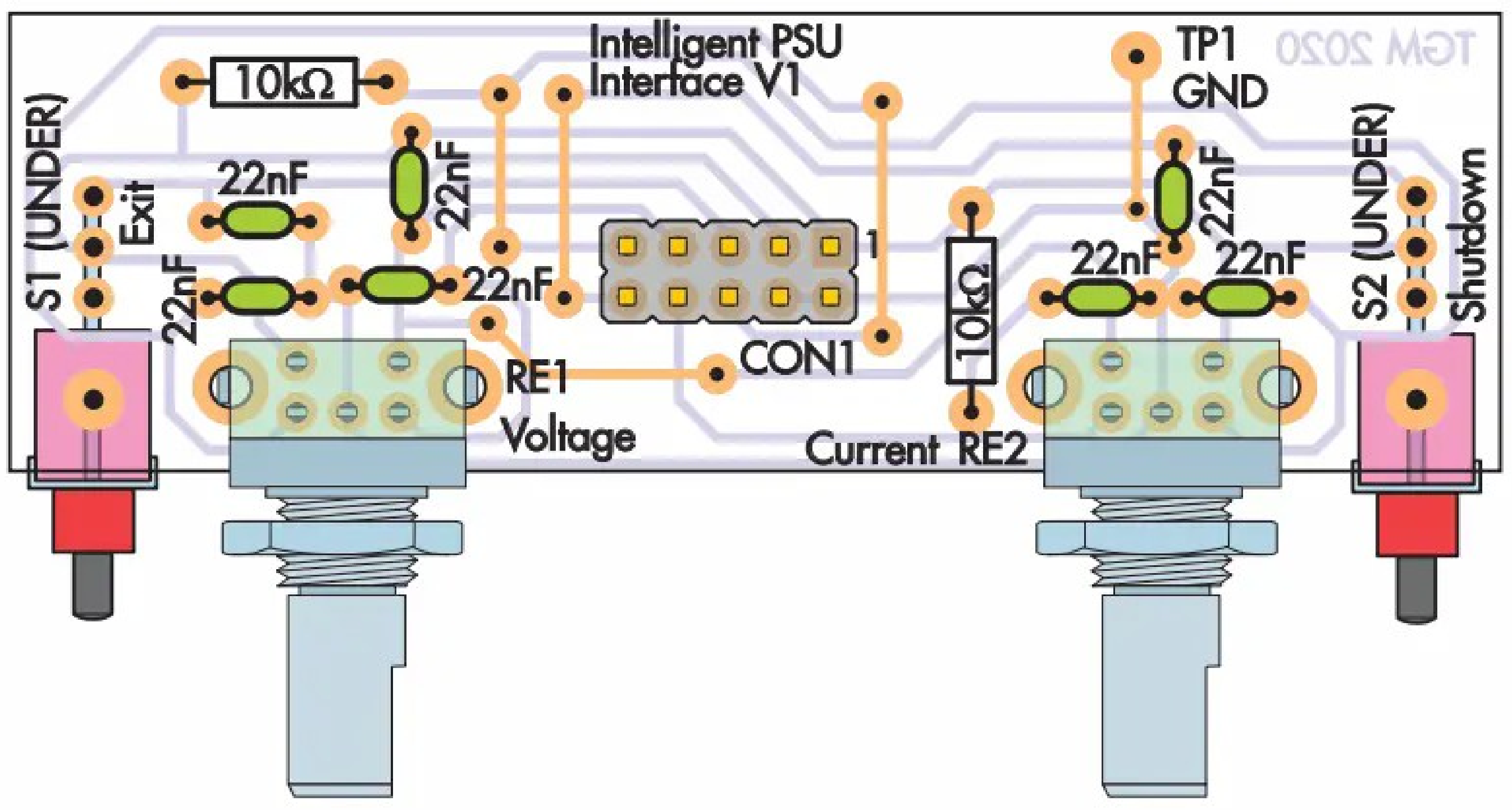

10kΩ
Intelligent PSU Interface V1
TP1
GND
TGM 2020
S1 (UNDER)
Exit
22nF
22nF
22nF
22nF
22nF
22nF
22nF
1
CON1
RE1
Voltage
10kΩ
Current
RE2
S2 (UNDER)
Shutdown

Fig.14: this simple front-panel board carries the two rotary encoders and two pushbuttons used to control the Supply, plus some debouncing components and pull-up resistors. It connects to the control board (shown in Fig.13) via a 10-way ribbon cable with DIL IDC connectors at each end.

Metalwork

The heatsink used is an Altronics H0545 300mm diecast aluminium type, with the final four fins cut off, as shown in Fig.16. This is to leave room for the power connector and fuse on the rear panel of the recommended case. It might seem an odd thing to do to a perfectly good heatsink, but it is otherwise ideal for the job, just a tad too long!

Cutting the heatsink is a 10-minute job using a hand-held hacksaw and a liberal dose of elbow grease. While it might look intimidating, no special tools are required. Clamp the heatsink to a workbench with cardboard protecting its surface and patiently work at it. Finishing off with a file will deliver you a neat result.

We taped it up and applied a quick spray of black paint to the cut section, but you don't have to do that.

There are six mounting holes to drill to 4mm, and ten mounting holes for regulators and brackets. We drilled and tapped these to M3 x 0.5mm. We have laid the PCB out so that the mounting holes are between the fins; if you do not have an M3 tap, you can simply drill these to 3mm and use long screws and nuts to mount the power devices straight through the heatsink between the fins.

Note how the mounting holes run along the top and bottom edges of the heatsink. All power device mounting holes are along the middle of the heatsink. In addition to the regulator ICs, the diode bridges are mounted to the heatsink, and there is a bracket in the middle of each regulator PCB.

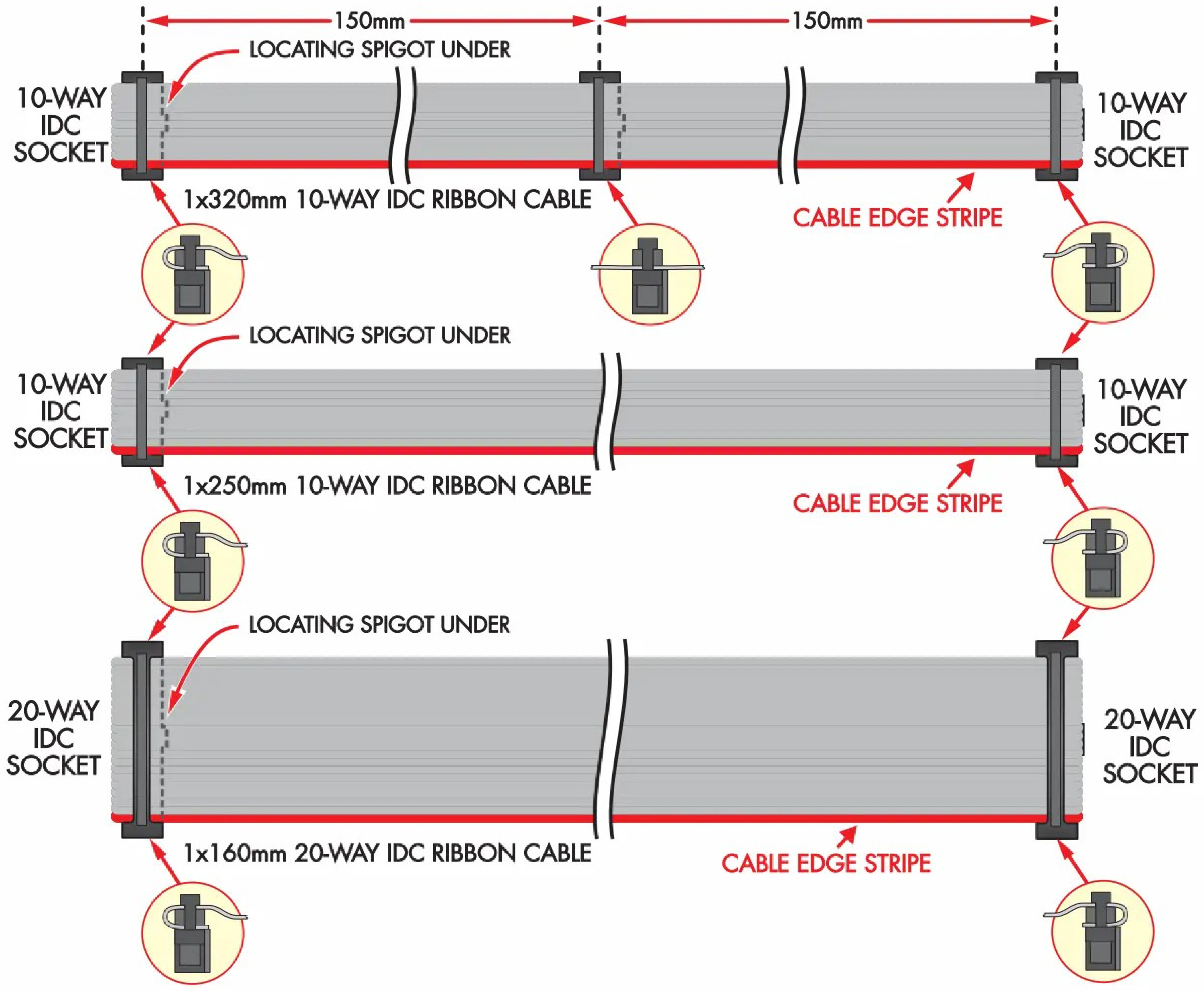

150mm
150mm
LOCATING SPIGOT UNDER
10-WAY IDC SOCKET
10-WAY IDC SOCKET
1x320mm 10-WAY IDC RIBBON CABLE
CABLE EDGE STRIPE
LOCATING SPIGOT UNDER
10-WAY IDC SOCKET
10-WAY IDC SOCKET
1x250mm 10-WAY IDC RIBBON CABLE
CABLE EDGE STRIPE
LOCATING SPIGOT UNDER
20-WAY IDC SOCKET
20-WAY IDC SOCKET
1x160mm 20-WAY IDC RIBBON CABLE
CABLE EDGE STRIPE

Fig.15: these are the three IDC cables you will need to make up to connect the boards. The 10-way cable with two plugs connects the control board (Fig.13) to the front panel (Fig.14), the other 10-way cable with three plugs connects the control board to the regulator board(s) (Fig.10) and the 20-way cable connects the control board to the LCD screen module.

Mounting the regulator modules to the heatsink requires a little care. Our approach was to fix the mounting bracket, insert all the power devices into their PCB pads and jiggle it around to get them aligned. We then screwed them loosely into their mounting holes and soldered their leads, as follows:

1. Install and screw down the mounting bracket in the middle of the PCB.
2. Bend the leads on the MC34167 to ensure that the device will mount flush to the heatsink. This device is a relatively tight fit.
3. Do the same with the two LM1084IT-3.3 devices, ensuring that you get them to about the right height.
4. Using silicone insulators, insulating bushes, flat washers, shakeproof washers and 16mm M3 screws, loosely mount the power devices to their locations on the heatsink. It is best to do this with the heatsink flat on a desk and the regulator module facing upwards.
5. Tack solder on one pin of each device.
6. Where there is a misalignment, reflow the solder on the offending pin to adjust it.
7. Secure the MC34167, then the LM1084IT-3.3 devices. You can access the mounting screw for the MC34167 through the gap between the 4700uF capacitors.
8. Once everything is aligned and there is no stress on the PCB, gently tighten all the mounting screws. Watch out that tightening the screw does not twist the device around, and make sure you don't overtighten them.
9. Now solder all the pins.
10. Mount the bridge rectifier on the heatsink now. It should already be wired to the PCB.
11. While you have the boards in this location, attach two 15mm long M3 threaded standoffs to the regulator module using 6mm M3 screws, flat and shake-proof washers. This will ensure it sits neatly on the desk.

When mounting the MC34167, it is easiest to stand the heatsink on end, slip the regulators into their holes and get the insulator in the right spot. Then using long nose pliers, line up the screws with the insulating bush and washers in the hole so you can do it up. Repeat this process for the LM1084 regulators, then the other regulator module (assuming you're building two).

Now put your multimeter on a high ohms range (eg, 20MΩ) and check the resistance between the heatsink and the tab of the TO-220 devices. There should be an open circuit in each case. If not, remove the device and check what has gone wrong; check in particular for burrs on the screw hole in the heatsink. This process is repeated for the second module.

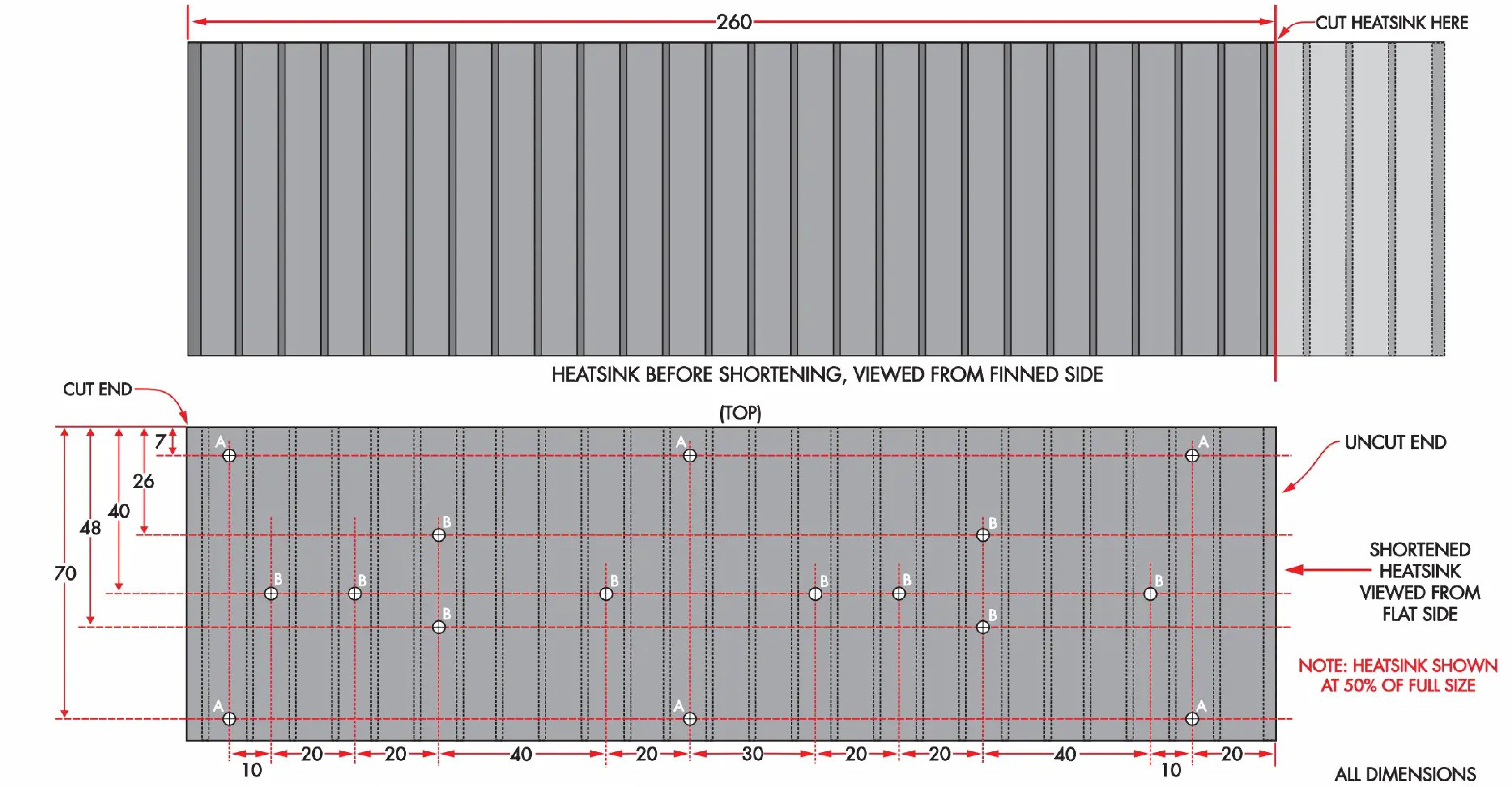
Initial testing

Connect the 20-way cable between the control board and LCD (being careful to line up pin 1 at both ends). Also attach the 10-way cable with two plugs between the control board and front panel board (the same comment applies) and the other 10-way cable between the control board and the one or two regulator boards.

You can make some initial checks at low power and without mounting anything to the heatsink. Just don't draw high currents!

Install jumpers on JP1 & JP2. If you are using only one module, select channel one only. For dual rails, select channel one on one board and two on the other. If you are not using our microcontroller-based control board, you do not need to install these and should not have loaded the DAC, ADC or opto-isolators.

During this testing, if it has not been mounted to the heatsink yet, make sure that the bridge rectifier


260
CUT HEATSINK HERE
HEATSINK BEFORE SHORTENING, VIEWED FROM FINNED SIDE
(TOP)
CUT END
UNCUT END
7
26
40
48
70
A
B
SHORTENED HEATSINK VIEWED FROM FLAT SIDE
10
20
20
40
20
30
20
20
40
10
20

NOTE: HEATSINK SHOWN AT 50% OF FULL SIZE

ALL DIMENSIONS IN MILLIMETRES

HOLES A: 3.0mm DIAMETER FOR HEATSINK MOUNTING HOLES B: 2.5mm DIA. & TAPPED M3, OR 3.0mm DIAMETER (FOR ATTACHING DEVICES)

Fig.16: this heatsink drilling pattern suits two regulator modules. The holes marked "A" are for mounting the heatsink to the case, while the two sets of holes marked "B" are for attaching the PCB-mounted semiconductors and bridge rectifiers on each module to the heatsink, plus a bracket to prevent the heatsink/PCB assembly from flexing too much. If you can't tap the holes for M3, they are positioned between the fins, so you can drill through and use long machine screws, washers and nuts.

can't short anything out. Either place it somewhere safe or wrap it in insulating tape.

Initial testing can be done by injecting ±15V DC into the board. Still, if you don't have a suitable dual supply (maybe that's why you're building this one?), you can instead solder a 10Ω 5W resistor across a blown M205 fuse and use this in place of the onboard fuse while applying about 24V AC to the input terminals.

If you have the dual DC supply, connect +15V to the rectifier side of the fuse, the external power supply ground to a ground point, such as the "-" on the bridge rectifier, and -15V to the large via just next to the 3300µF capacitor. You can solder a piece of wire into this via and clip a lead to it.

Switch on and allow it to settle. The current draw should be less than 200mA. Check for the following voltages:

- +12V (typically closer to 11.5V) on pin 2 of REG1 (LM317). This is also on the cathode of D2, just below the regulator. This should be within a volt of the expected value.
- +5V (5.1V actual) on pin 2 of REG2 (LM317). This is also the cathode of D10, just below the regulator. This should be within 0.5V of the expected value.
- -4.5V (-4.5V actual) on pin 3 of REG4 (LM337). This is also on the anode of D17 next to the regulator. This should be within 0.5V of the expected value.
- +5V on LK1, generated by REG3 (LM2575). This should be within 0.5V of the expected value.

We left a 4mm gap between the big capacitors to allow a screwdriver to get to the tab of the TO-220 devices. It is tight, but enough for a standard Philips screwdriver. It's easiest to start by holding the screw with long-nose pliers.

If any of these are outside of the expected ranges, check the following:

- Is the supply current high? Feel for components getting warm.
- Look for solder bridges.
- Check that the electrolytic capacitors are in the right way around.
- Check that the regulators, diodes and ICs have been installed with the correct orientations.

Assuming that's all OK, verify that the pre-regulator is working. With no controller connected, the output should be set to 0V automatically. That means the pre-regulator should be producing around 5V. You can probe this on the output side of the 220µH inductor (the pin away from the MC34167). The exact voltage is not critical, but it should be between about 5V and 6.6V.

If this is not as expected, check the following:

- If you have an oscilloscope, set it to measure 5V/division and probe pin 2 of the MC34167. You should see some serious switching waveforms. It might not be switching at 72kHz, as the regulator will be unloaded and possibly running in discontinuous mode.
- Check for solder bridges in the switchmode area. If the output is in the range of 0-0.5V, check for shorts around the schottky catch diode (D3).
- Check the voltage on pin 1 of the MC34167. It should be close to 5V.

Remainder of case

Once you have finished preparing the heatsink, move on to the rear panel.

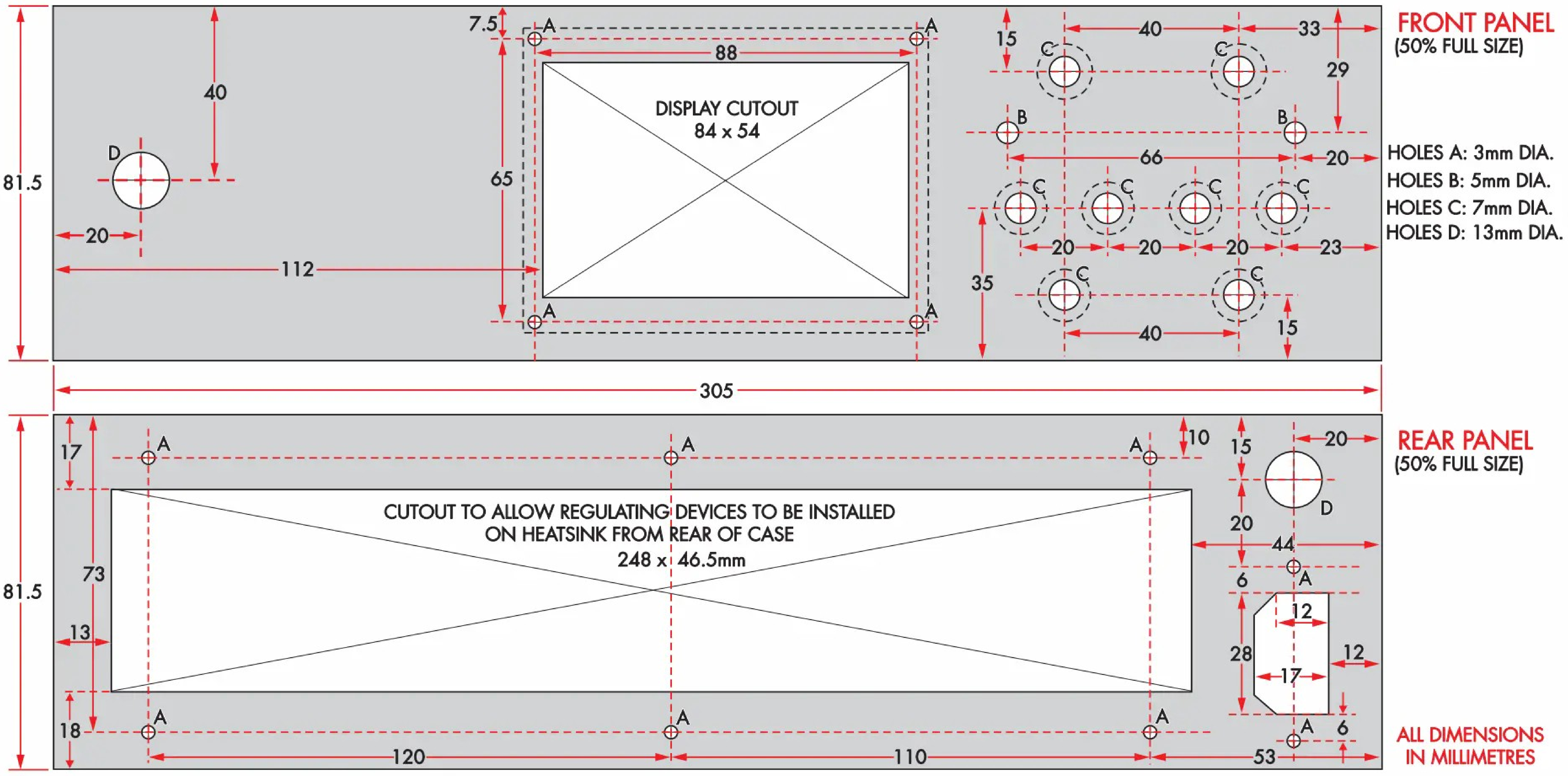


Fig.17: the case's front and rear metal panels need to be drilled and cut as shown here. The large rectangular opening at the rear allows the regulator PCBs to be admitted into the case after being attached to the main heatsink. The main heatsink then bolts to the rear of the case via the six holes marked "A" around the cutout. See the text for advice on how to cut the large holes.

Remove it and drill and cut the holes, as shown at the bottom of Fig.17.

To make the rectangular hole and the D-shaped hole for the mains socket, we drilled large holes in each corner of the cutouts and used a handsaw with a metal blade to cut along the outlines. Other approaches would be to use a jigsaw with a metal blade or a rotary tool (eg, a Dremel) with a metal cutting disc.

In all cases, be somewhat careful as the material in the recommended case is aluminium, and you will easily bend it once cut. Once this is mounted to the heatsink, it will regain its strength. We have used a large hole to allow the complete heatsink assembly, with regulator modules, to slip in from the back.

Now present the heatsink and regulator modules to the rear panel. The assembly should slip through the large cutout, and the mounting holes in the heatsink should line up with those in the rear panel.

If there is a minor misalignment, simply drill the offending holes to 4mm or so. Fix these using 16mm M3 machine screws, flat and star washers and nuts.

Finally mount the IEC panel male socket and fuse holder. The IEC socket is fixed using 16mm M3 screws, flat and star washers and M3 nuts.

The base needs a few holes drilled, shown in Fig.18. We have provided locations for the regulator module mounting holes. Still, given the variability in how you have mounted the PCB to the heatsink, you will be better off putting some masking tape at the identified locations, installing the rear panel with regulator modules mounted and marking the exact locations before drilling those holes.

While you are there, mark and drill the remainder of holes in the baseplate: the Earth post near the IEC connector, the toroidal transformer mount, the two holes for the terminal block and four holes for the control PCB.

Cut Presspahn or similar insulating material and place this under the terminal block. Mount and secure the terminal block with 16mm M3 machine screws and nuts. Remove the paint around the Earth post and mount a 16mm M3 screw with a shake-proof washer and M3 nut. Reserve another shakeproof washer, M3 nut and solder lug to attach the mains Earth wire.

Mount the control PCB using four

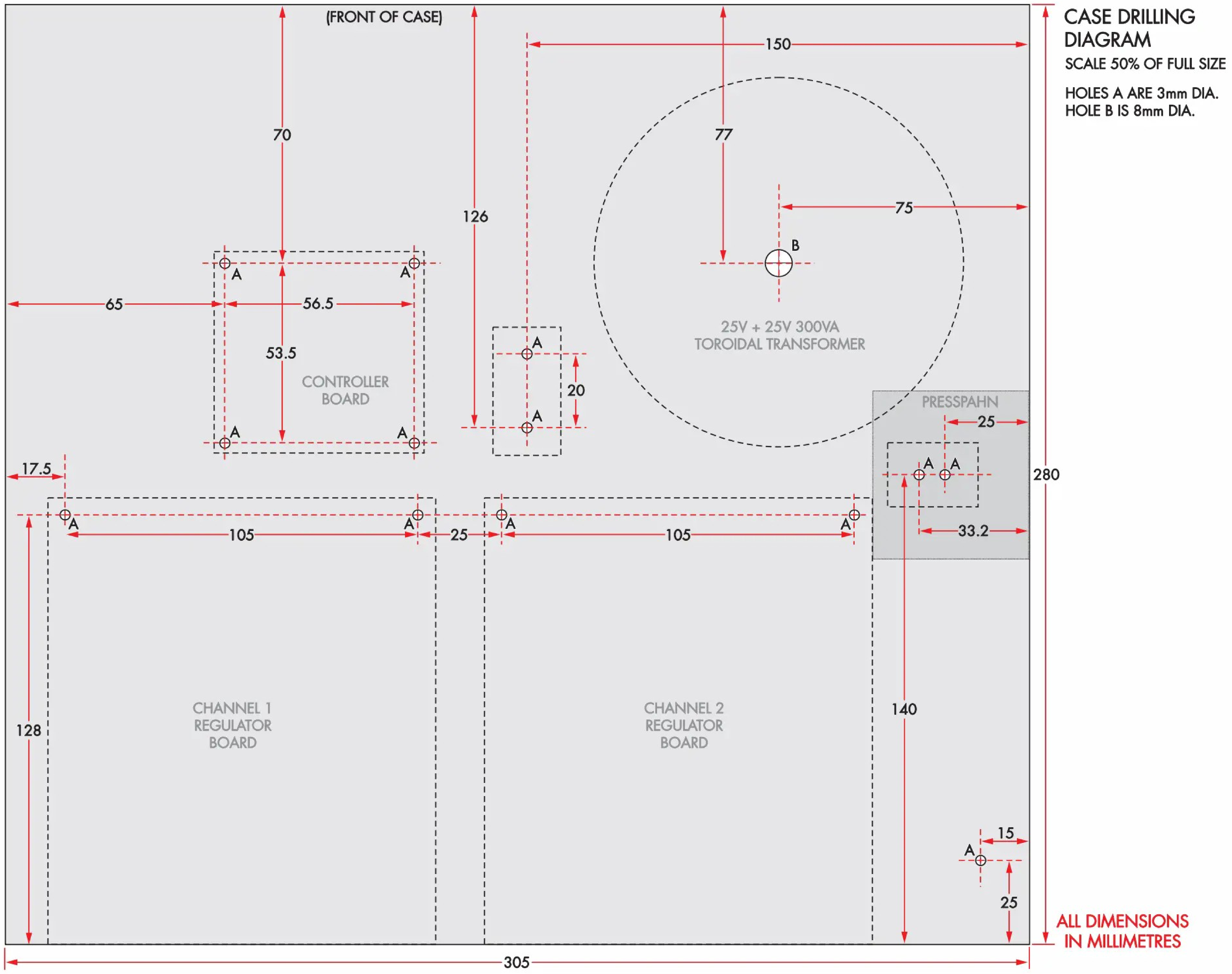

(FRONT OF CASE)
CASE DRILLING DIAGRAM
SCALE 50% OF FULL SIZE
HOLES A ARE 3mm DIA.
HOLE B IS 8mm DIA.
150
70
77
126
75
B
A
65
56.5
53.5
CONTROLLER BOARD
20
25V + 25V 300VA TOROIDAL TRANSFORMER
PRESSPAHN
25
17.5
280
105
25
105
33.2
CHANNEL 1 REGULATOR BOARD
CHANNEL 2 REGULATOR BOARD
128
140
15
25
305
ALL DIMENSIONS IN MILLIMETRES

Fig.18: this shows where you need to drill holes in the bottom of the case to mount the regulator modules, transformer, control board and terminal blocks for terminating both the low-voltage and high-voltage windings on the main transformer. It's best to check where exactly your regulator modules sit when mounted in the case to ensure their holes are drilled accurately.

15mm Nylon standoffs and 6mm M3 screws with fibre washers under the heads. This assures good separation of the mains Earth from the control PCB.

Front panel

The front panel needs a few holes and a large cutout for the LCD. Read through this section and check the measurements of your parts before cutting. This front panel will be right there on your workbench for a long time, so make it neat! The details are in Fig.17.

Drill and deburr the holes as shown. For the LCD cutout, provided you are using the acrylic panel with paint to hide the cutout, you can err on the large side for the hole, as the paint will hide the hole you cut.

Mount the switches and connectors, then attach the control PCB using the rotary encoders to secure it to the front panel.

That just leaves the LCD screen. Mounting the display in a manner that insulates the LCD bezel from the case is fiddly. We found that some panels supplied have the LCD panel ground pin connected to its metal bezel.

The display has 2.5mm mounting holes, so unless you have plenty of 16mm M2.5 machine screws and nuts on hand, drill these to 3mm. Clean up any burrs that result.

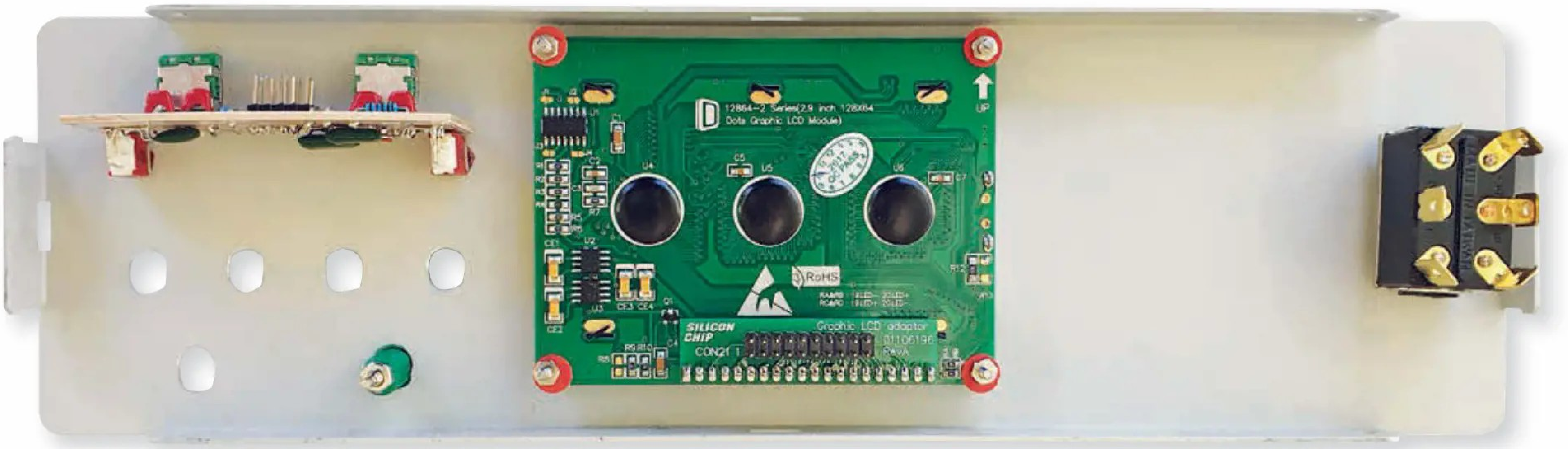
Now take the 3mm acrylic sheet and cut it as shown in Fig.19 (or purchase a laser-cut version from the Silicon Chip Online Shop).

The mounting holes on our display were 88mm by 65mm apart. We carefully drilled mounting holes through the acrylic as shown, and came up with an arrangement that mounts the acrylic to the front panel and holds the LCD to the acrylic rather than the LCD bezel touching the case.

You could be lucky, and your LCD bezel may be isolated from the case – but please do not assume this to be the case, as we want isolation between the power supply and mains Earth. This arrangement gives you good clearance.

To make things neat, after cutting and adjusting the cover to fit the LCD and case, we masked off the inside of the cover as shown, then spray painted it black. Once the masking was taken off, we had a neat black shadow line that hides the cut in the case and gives a professional appearance.

The assembly of the acrylic cover, the case and LCD is as shown. The acrylic cover mounts to the metal case



Secure the acrylic bezel to the case with the M3 screws, then slip the LCD screen onto their shafts and tighten it up against the acrylic with insulators under the nuts.

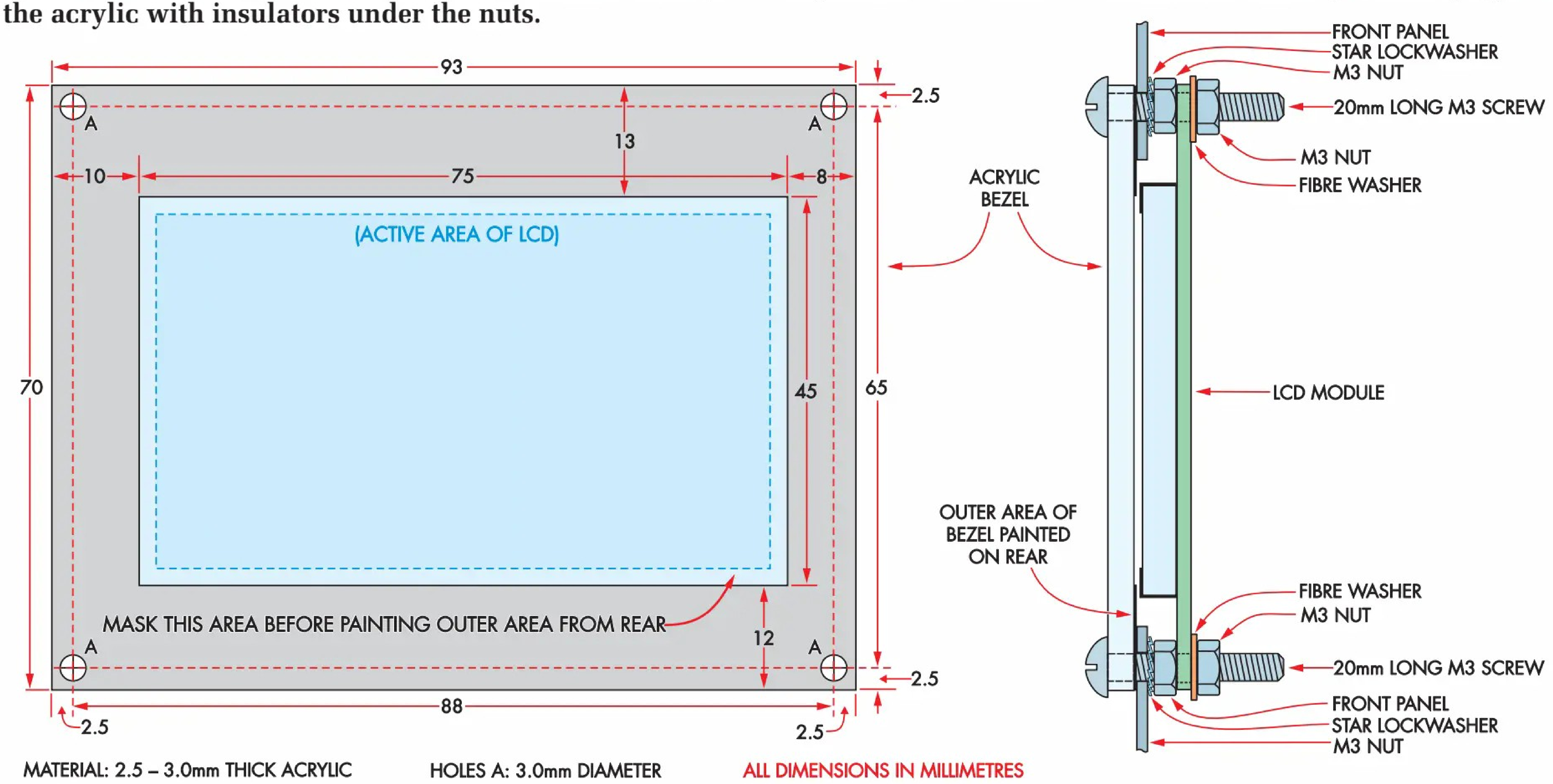
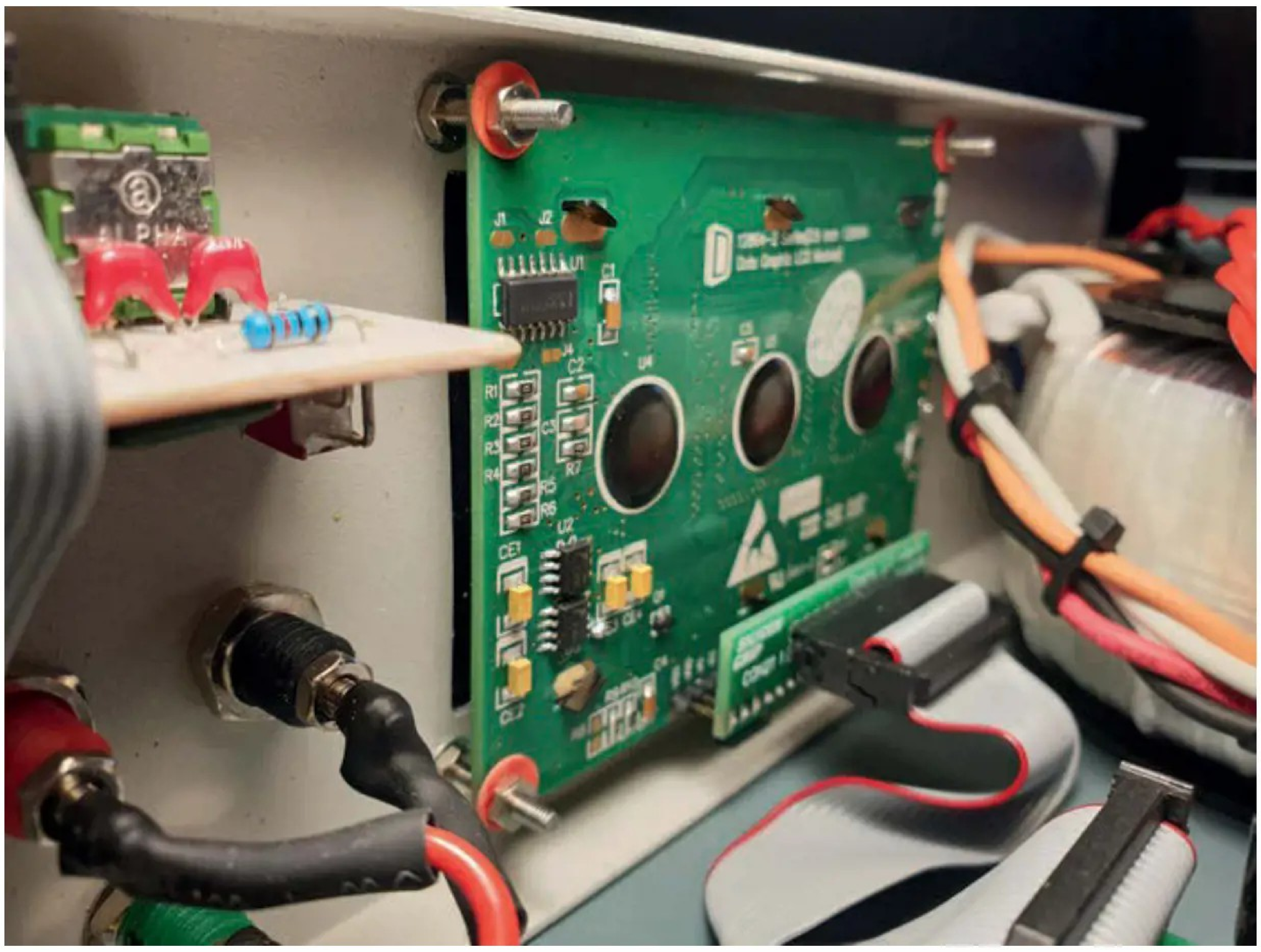

93
2.5
A
A
13
10
75
8
(ACTIVE AREA OF LCD)
70
45
65
MASK THIS AREA BEFORE PAINTING OUTER AREA FROM REAR
12
A
A
2.5
88
2.5
2.5
ACRYLIC BEZEL
OUTER AREA OF BEZEL PAINTED ON REAR
FRONT PANEL
STAR LOCKWASHER
M3 NUT
20mm LONG M3 SCREW
M3 NUT
FIBRE WASHER
LCD MODULE
FIBRE WASHER
M3 NUT
20mm LONG M3 SCREW
FRONT PANEL
STAR LOCKWASHER
M3 NUT
MATERIAL: 2.5 – 3.0mm THICK ACRYLIC
HOLES A: 3.0mm DIAMETER
ALL DIMENSIONS IN MILLIMETRES

Fig.19: as it's tough to make a clean and rectangular cutout in the metal panel for the LCD, we designed this plastic bezel to cover up the screen surround. You can order a laser-cut clear bezel with these dimensions from our website (at the same time you order the PCBs etc), but you will have to paint the outer area yourself.



using M3 (or M2.5) machine screws. The hole in the case is slightly larger than the bezel on the LCD, so the LCD can then be secured using four more machine screws and fibre washers.

At this point, all parts of the case should be cut and drilled, and the PCBs mounted and ready to wire up. Refer now to the wiring diagram, Fig.20. There are three control cables, which are routed as shown.

Move on to the front panel and low-voltage wiring. We have included the wiring of the bridge rectifiers for completeness, although this should have been addressed while building the regulator module(s).

When connecting the wiring to the front panel, solder two 100nF 50V capacitors across the pairs of output terminals. These reduce the noise from the switch-mode section of the PSU on the output.

Also install 10nF capacitors from each of the negative outputs of the two channels to mains Earth. Without this, the capacitive coupling through the mains transformer will induce substantial floating voltages on the channel outputs.

Now it's time to mount everything else in the case. Start by locating the mains transformer and wiring the outputs to the terminal strip as shown. We have used colour coding to match the Altronics M5525C 25 + 25V transformer. Check yours before proceeding as an error in this part of the circuit would not be good.

We used 13cm and 25cm pairs of 7.5A-rated cables (red) from the terminal strip to the AC inputs of the regulator modules. Twist these to keep things neat, and tuck them away between the boards.

We have made provision on the front panel for mains Earth access at each of the outputs. In some circumstances it can be handy to connect one of the outputs to Earth, but other times it is not desired. These mains Earth points need to be wired back to the front panel Earth lug using green/yellow striped mains-rated wire.

The final connections to the front panel are the outputs from each of the two regulators, made with red and black (or blue) 7.5A-rated cable. Make sure these are secure. Ours were 24cm and 34cm in length after twisting together and terminating.

While you are here, it is worth terminating the mains side of the toroidal transformer to the terminal block.

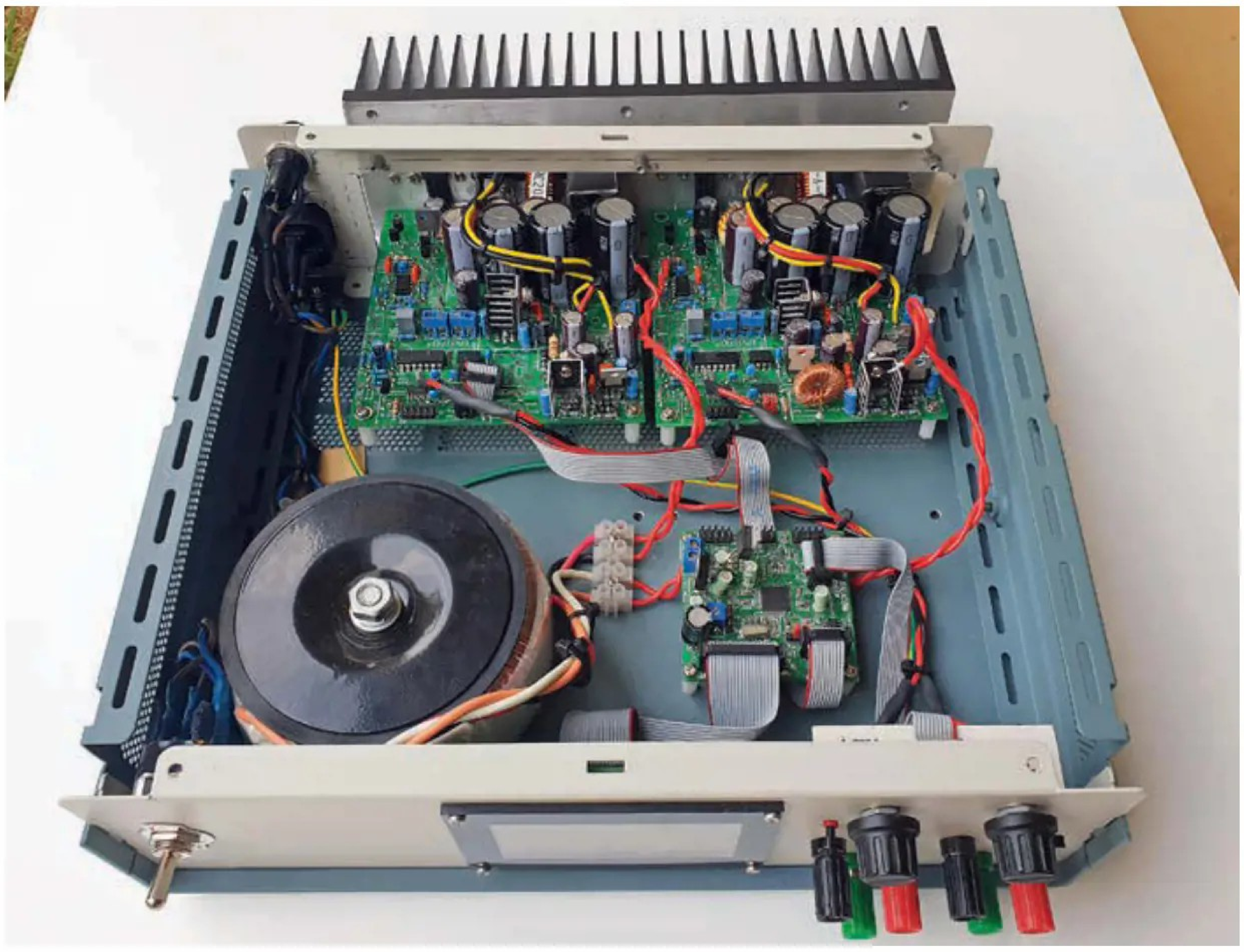
Mains wiring

Use mains-rated cables for all wiring. Be careful to check this and if you have someone you trust, get them to look over it too.

With the fuseholder and IEC socket installed, fit the Earth screw at the rear of the case. Make sure you scrape the paint off the case in the bolt area and use star washers on top and bottom. Do it up securely.

First, connect the Earth wire from the IEC socket to the Earth lug using a solder lug and heatshrink tubing, to keep things tidy. Make sure this is long enough that it will not be strained.

Next, run all the Earth wires from the main Earth screw in the base to the panels shown in Fig.20, including



The Regulator modules mounted to the heatsink slide straight into the case. We used nutserts on the rear panel, allowing us to screw the heatsink straight into them, making assembly a breeze. If you build a reasonable amount of sheet metalwork, do yourself a favour and buy a nutsert tool!

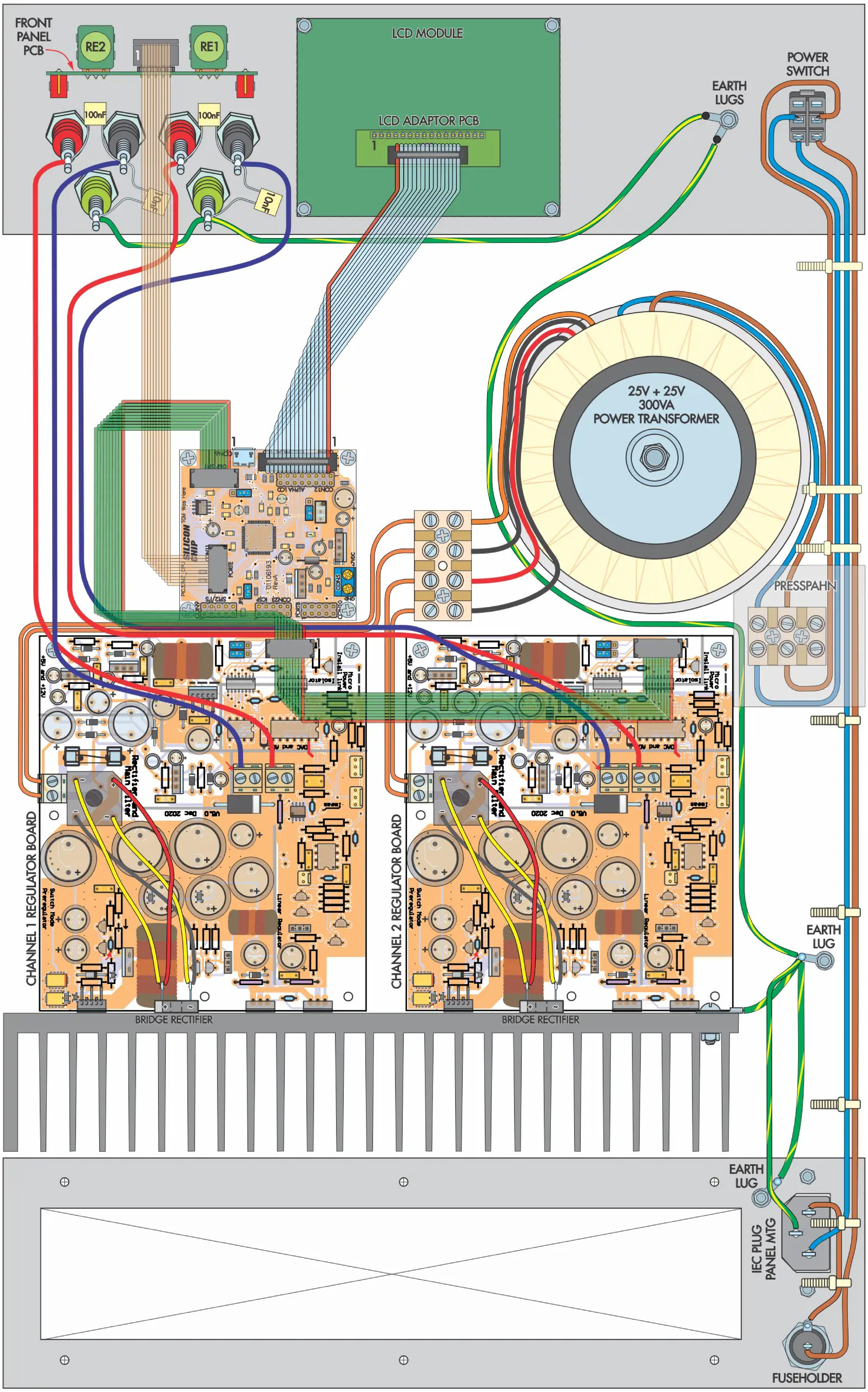
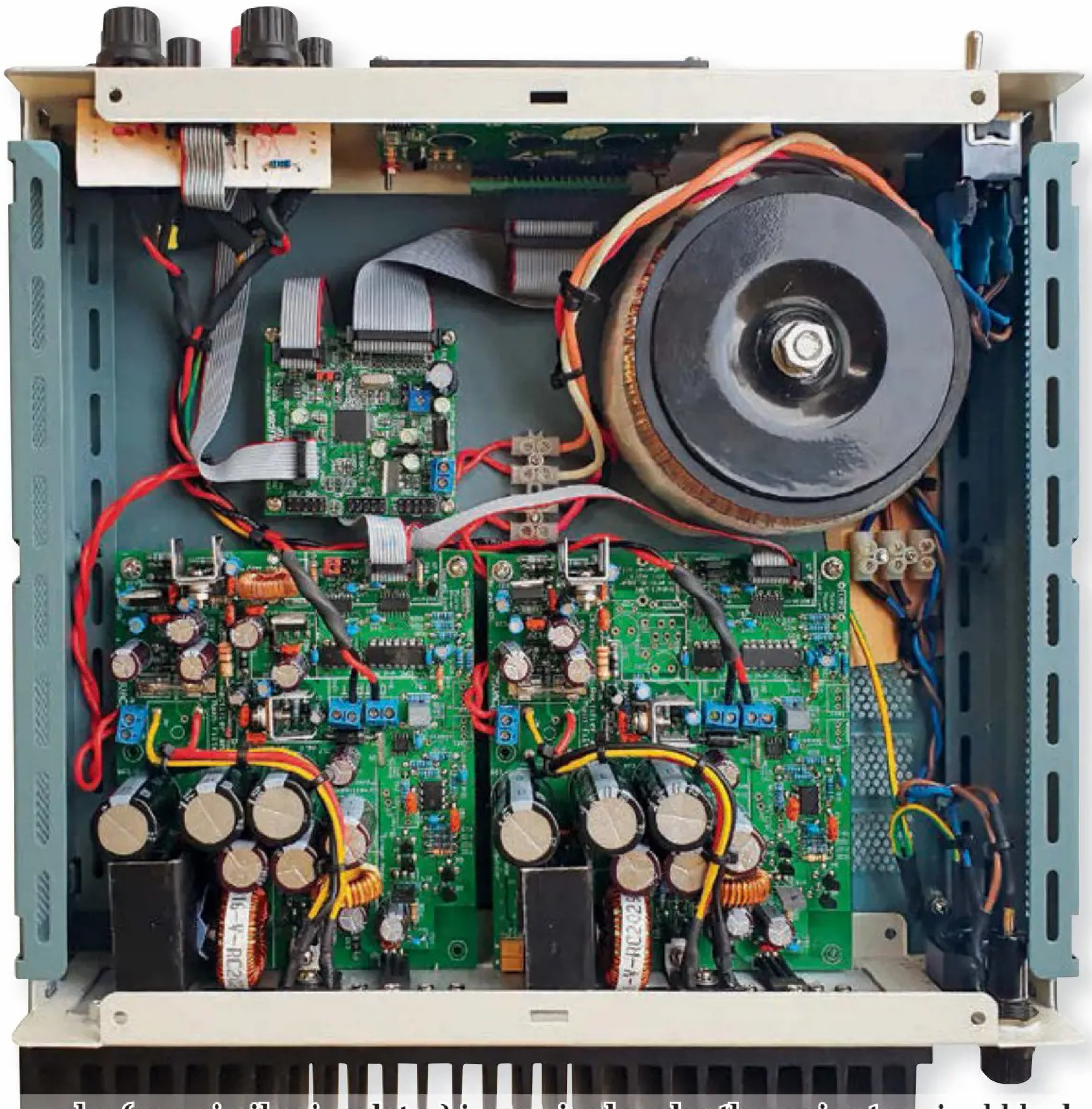

FRONT PANEL PCB
RE2
RE1
1
100nF
100nF
10nF
10nF
LCD MODULE
LCD ADAPTOR PCB
1
EARTH LUGS
POWER SWITCH
FRONT PANEL (INSIDE VIEW)
INTELLIGENT POWER SUPPLY WIRING DIAGRAM
(50% OF FULL SIZE)
25V + 25V
300VA
POWER TRANSFORMER
BASE OF CASE
PRESSPAHN
CHANNEL 1 REGULATOR BOARD
CHANNEL 2 REGULATOR BOARD
BRIDGE RECTIFIER
BRIDGE RECTIFIER
EARTH LUG
EARTH LUG
IEC PLUG PANEL MTG
REAR PANEL (INSIDE VIEW)
FUSEHOLDER

Fig.20: this wiring diagram should make clear all the connections needed to complete the Supply. Ensure the Earth lugs are making good contact with the bottom of the case and the rear panel; if necessary, clean off any paint or coating around their mounting holes and use shakeproof washers to ensure they 'bite' properly. All mains wiring must be properly insulated, including at the rear of the front panel power switch and the rear panel mains input socket.



Presspahn (or a similar insulator) is required under the mains terminal block for safety (shown along the right edge of the case). This photo shows the wiring in place. Make sure that all of the metal chassis panels are connected to mains Earth when assembled either via the securing screws or Earth wiring.

the heatsink (eg, using one of the existing bracket mounting screws to attach it).

Then using brown wire, connect the Active line from the IEC socket to the fuseholder, and from there to the power switch. We used a 6.3mm crimp connector here; you could solder it directly, provided you insulate the connection properly. Again, keep things secure, and use cable ties to ensure that, should any wire break or joint fail, the ends will be controlled and not create a hazard.

Using light blue wire, run the Neutral connection from the IEC connector to the power switch. Ensure that you connect the IEC input to the bottom (switched) pins on the power switch. This way, when the power is off, the unused switch terminals will be connected to the transformer, not the mains. For safety, put heatshrink tubing on the unused power switch pins anyway.

At this point, everything should be wired up and ready to go.

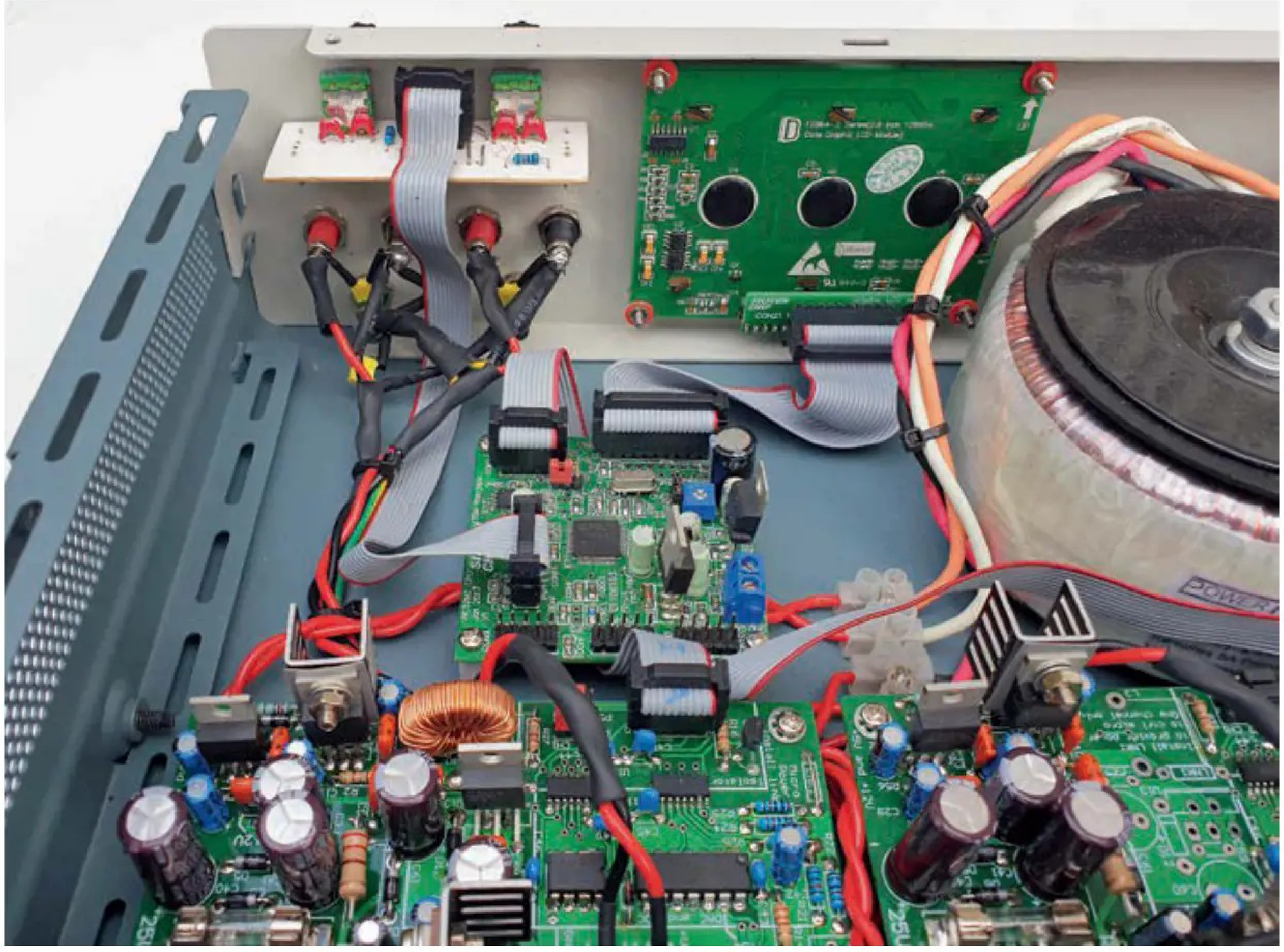
Calibration and use

First, make sure you have the CH1 and CH2 jumpers on! When you power the unit up initially, if the EEPROM contains garbage data, it will choose its own defaults to get things running. Do not rely on this as they might not be suitable for you!

You can now power the unit back up, and should be able to fully control and monitor voltages and currents from up to two regulator modules.

The initial setup procedure is:

1. Click the exit button to the lower left of the voltage set dial. This brings up the setup menu.
2. The voltage dial will allow you to select between three sub-menus: Track, Power and Cal. Enter the Track menu.
3. If you have a single output, N/A will be shown. Otherwise, select dual tracking or independent rails.
4. Enter the Power menu. For the number of rails, select single or dual-channel mode.
5. Set the absolute maximum current limit; this should be 5A in most cases. This can be set lower to limit current below that which the transformer VA rating allows; for example, if you are letting students loose with the Supply.
6. Dial up the maximum output voltage until the stated "required transformer voltage" matches your transformer.
7. Dial in the correct transformer VA rating. The recommended transformer is 300VA.
8. Enter the Cal menu and check the following as an initial starting point for both channels:
   - 8.1 Output offset measured at zero volts set = 0mV
   - 8.2 Set Correction Coefficient = 1.000
   - 8.3 Read Correction Coefficient Scale = 1.000
   - 8.4 Set Current offset = 0mA
   - 8.5 Current Correction Coefficient Scale = 1.000



Dual Hybrid Bench Supply, part two

Now check that the Supply generally works. You can set the voltages for channels 1 & 2 using the left-hand rotary encoder. To swap between them, push the dial and it will click. Channel 1 or 2 will be highlighted on the screen.

You can set current limits for channels 1 and 2 using the right-hand dial. Similarly to above, pushing the dial will swap between the channel 1 and 2 limits. By clicking either dial, you will save all settings.

Now perform calibration. Check that the calibration offsets are zeroed as described above, or else this procedure will be confusing:

1) Output offset

This sets the zero for measured voltage, taking out any offset. Set the Channel 1 voltage to 0.00V on the main menu, and measure the output voltage from the regulator module. Ours was -4mV; it should not be a large value.

Go into the CAL menu. The first screen says "Ch#1 Output Offset Measured" (see Screen 1). Adjust the voltage dial in the opposite direction to your reading until the output reading is close to 0V. An output voltage within a few tens of mV of zero is acceptable. If you have to dial in a significant value, check your build as this should not be required.

2) Voltage correction coefficient

This step sets the scale correction for output voltage, correcting for gain errors in the DAC and feedback network. Go back to the main menu and set the output to a high output voltage that you can measure accurately. For many meters, 19.99V is a good value. This will depend on the transformer you have selected.

Now go to the second calibration screen (Screen 2) and adjust the Voltage dial until you read 19.99V (or your chosen value) on your voltmeter. Our coefficient was 0.966, as shown. A value between about 0.85 and 1.15 would be reasonable, although it's likely to be in the range of 0.95 to 1.05. Do not worry about what the "Meas" voltage says on the main menu just yet!

3) Voltage reading correction coefficient scale

This step sets an ADC measurement correction to ensure voltage measurements displayed in the main menu are accurate. With the voltage still set to 19.99V, click on the Voltage READ correction coefficient scale menu. You will see the output voltage at the bottom of the screen as measured by the Regulator module for that channel – see Screen 3.

Adjust the Voltage dial until you get a reading of 19.99V (or your chosen value) on the bottom of the calibration screen. Our calibration factor was 1.037; values between 0.85 and 1.15 are reasonable.

4) Current reading offset

This step makes sure that you get current readings of 0mA when no current is flowing. With the voltage still set to any value, but no load connected to the power supply, click onto the Current Read Offset menu. At the bottom of the screen, you will see the current as measured by the Regulator module for that channel (Screen 4).

Now adjust the Voltage dial upwards until you read a current on the bottom of the calibration screen, then dial it back to get zero.

5) Current scaling coefficient

This step sets the calibration scaling for current measurements, so displayed currents are accurate. You need a dummy load for this test. Any high-power resistor will do; you can use quite a low voltage from the power supply, so two 1Ω 5W resistors in parallel will do for a short test. This keeps the dissipation to 4W per resistor.

Depending on your transformer setup, choose a current that is close to your maximum; say, 4A for a 5A unit. Check that the current limit is set above this value. If you cannot set it high enough, go back and check your transformer configuration.

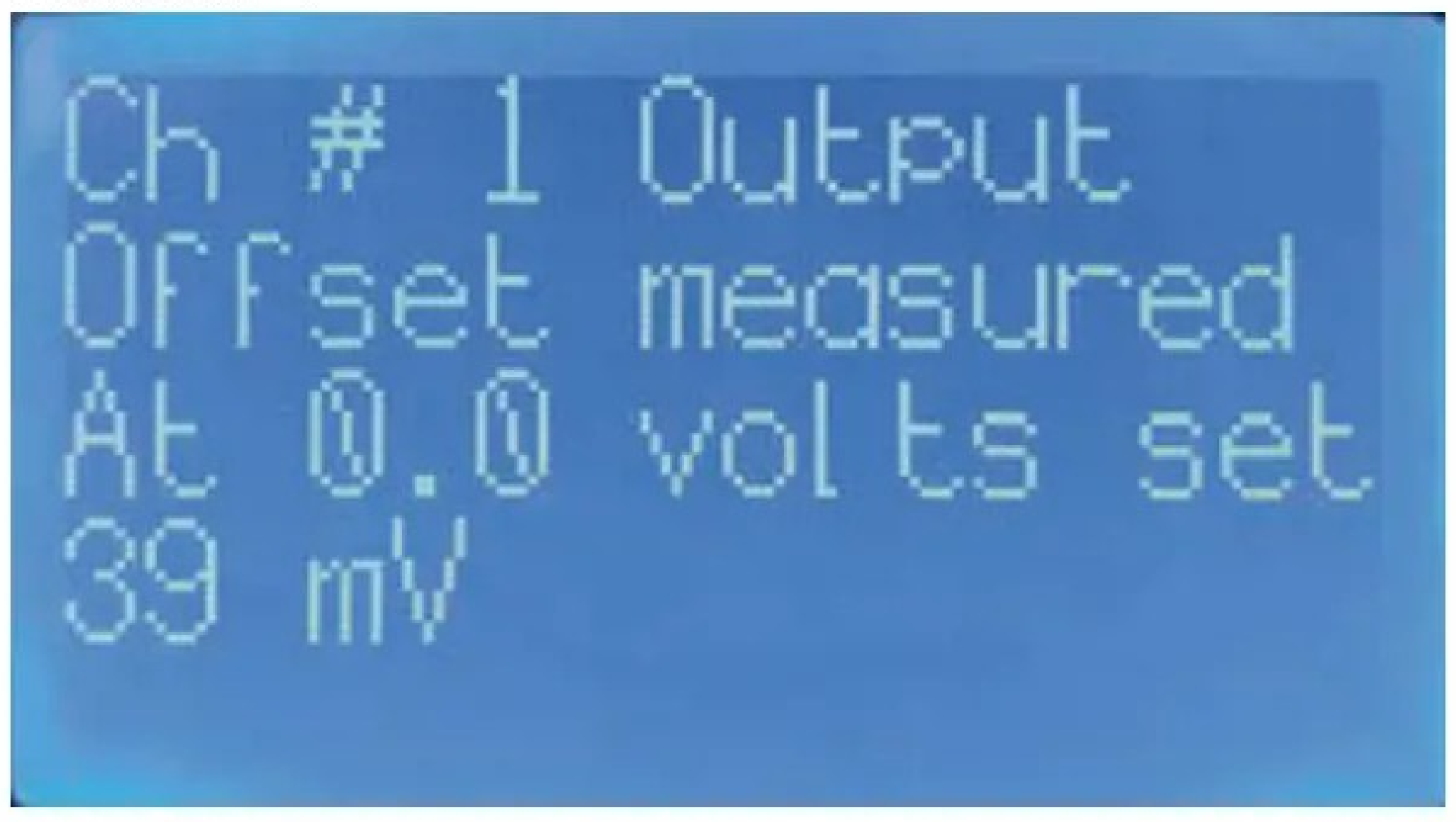
Now put an ammeter in series with the resistor and dial the voltage to achieve your target current. Next, adjust the Voltage dial until you see the correct current reported on the bottom of the calibration screen – see Screen 5. Then click on the EXIT button.

After that, press the voltage dial to swap channels and SAVE the calibration data.

Calibration is complete for channel 1, so repeat the whole procedure for channel 2.

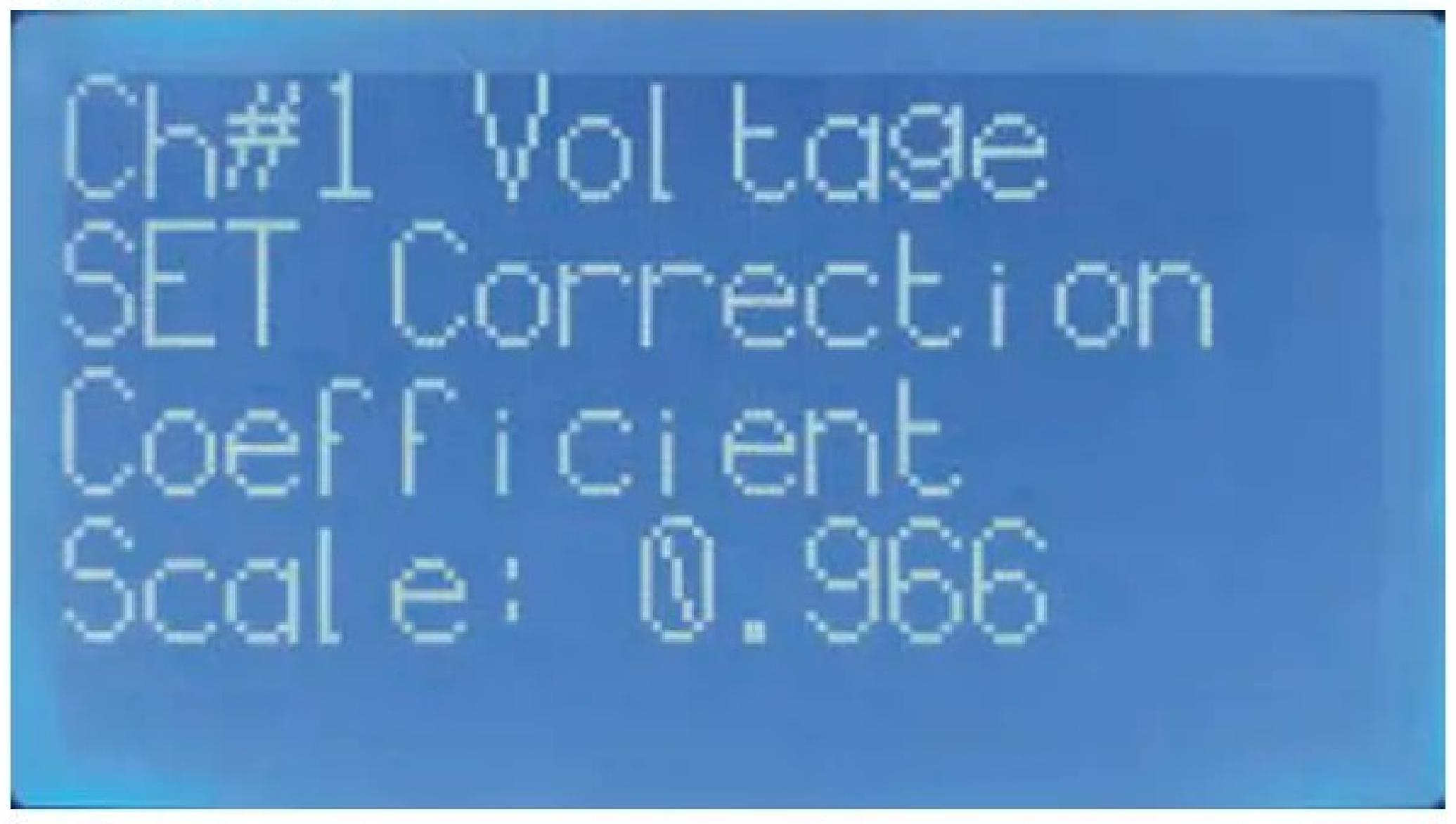
You should find that measured values are within 1% or so of the actual values. We have not attempted to make a laboratory-grade voltage source here, but the ADC we have chosen does have better than 0.1% resolution. Long-term precision will depend on the stability of the +5V internal voltage rail. Current measurement will be similar in terms of precision and stability.

Screen 1



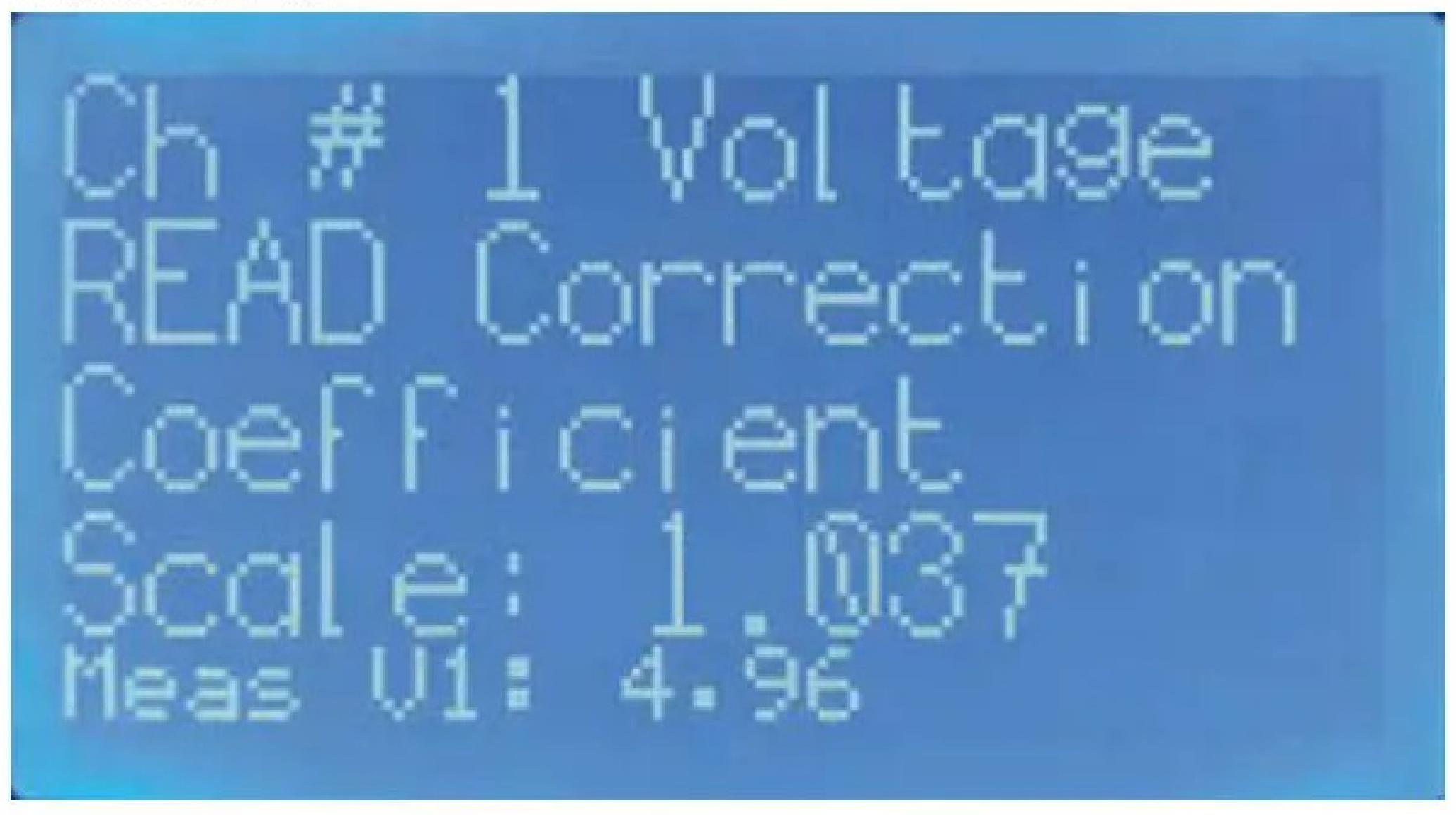
```
Ch # 1 Output
Offset measured
At 0.0 volts set
39 mV
```

Screen 2



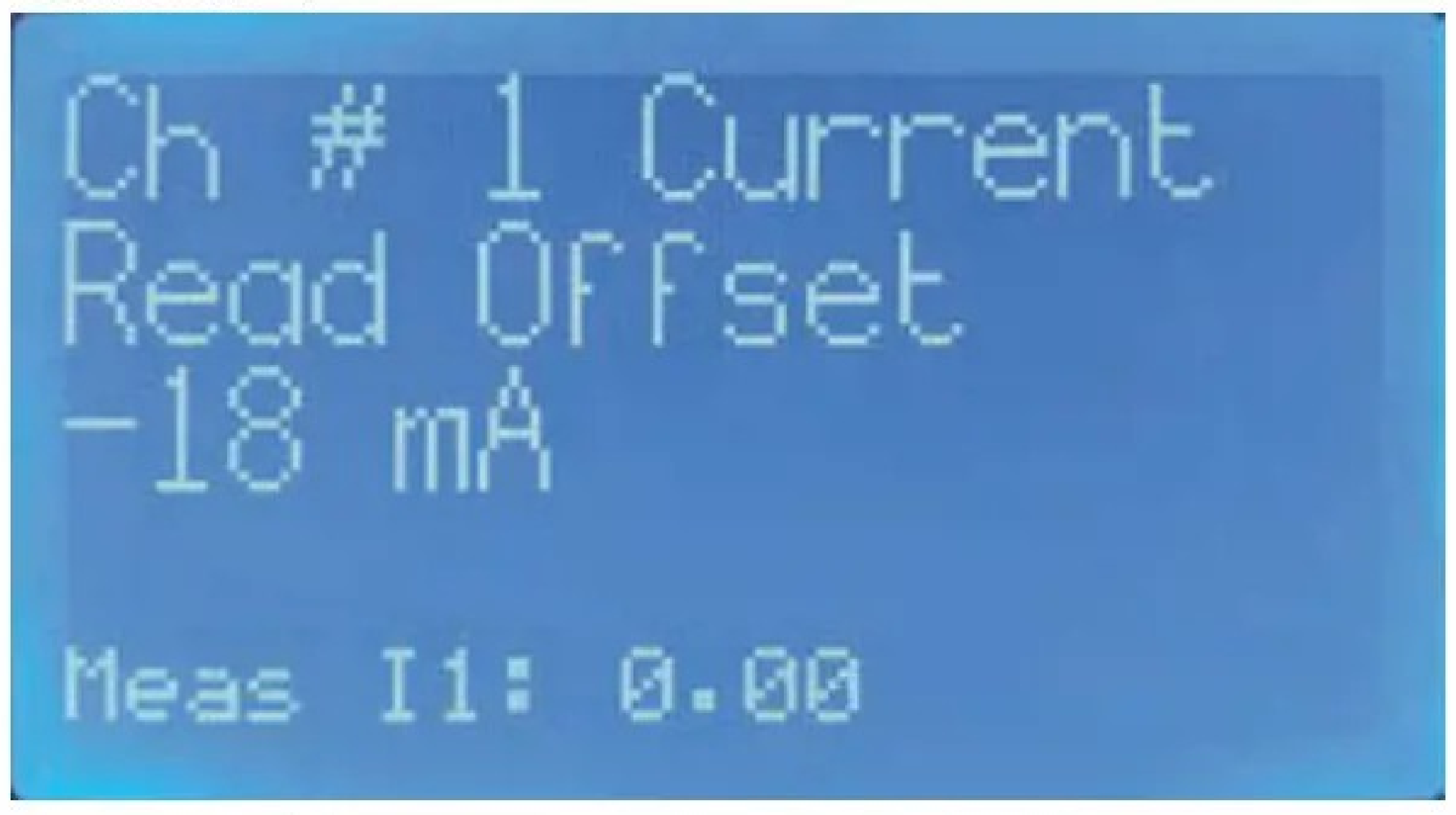
```
Ch#1 Voltage
SET Correction
Coefficient
Scale: 0.966
```

Screen 3



```
Ch # 1 Voltage
READ Correction
Coefficient
Scale: 1.037
Meas V1: 4.96
```

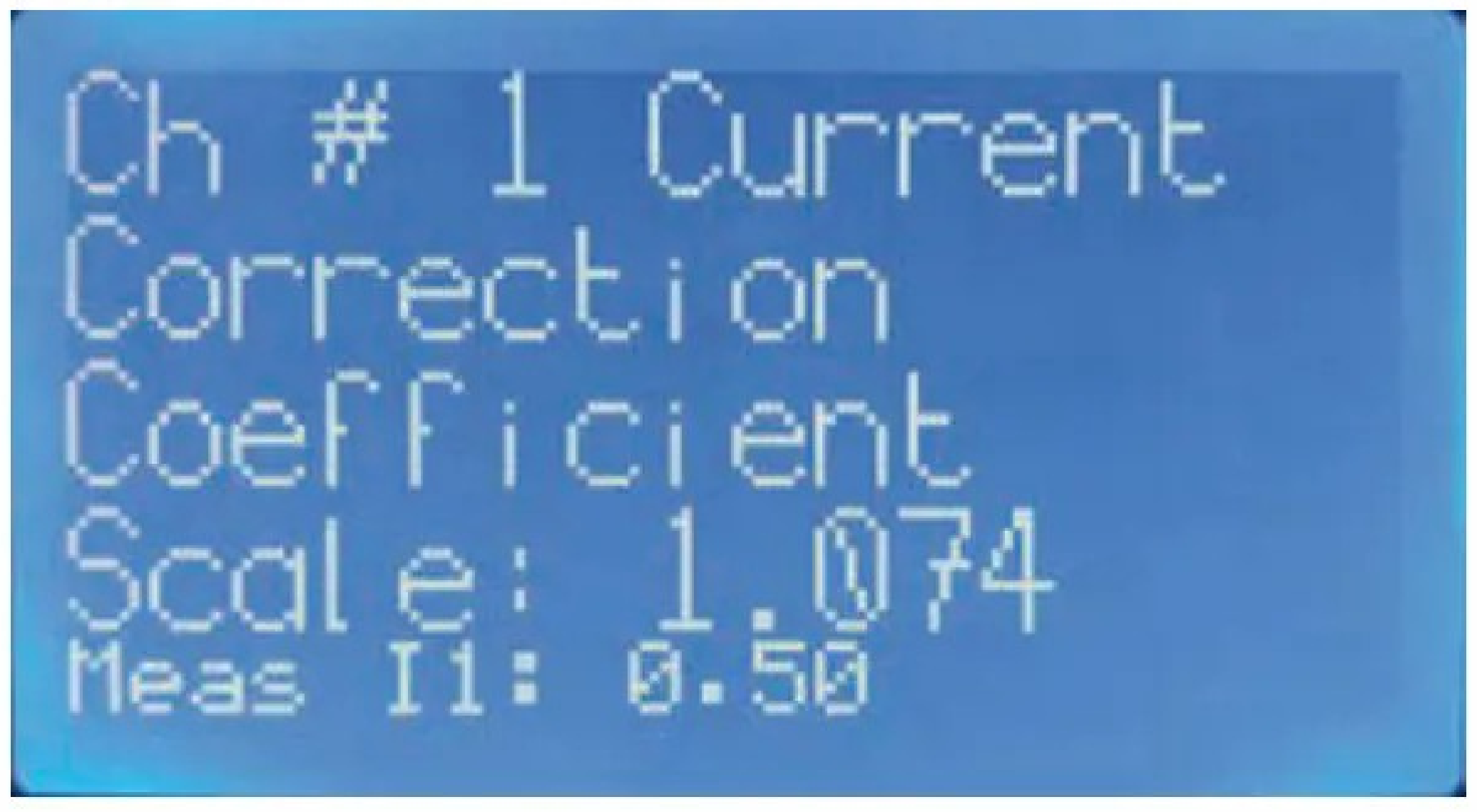
Screen 4



```
Ch # 1 Current
Read Offset
-18 mA

Meas I1: 0.00
```

Screen 5



```
Ch # 1 Current
Correction
Coefficient
Scale: 1.074
Meas I1: 0.50
```

You will notice that if the current output is within 5% of the limit current, we highlight the "I" symbol on the user interface. Similarly, if the output voltage is too low or high, we highlight the "V" symbol. There are headers for LEDs that you can wire to the front panel for over-current indication too, if that takes your fancy.

This completes the assembly and setup of the Intelligent Power Supply. We think this will be a valuable addition to most workbenches.

Practical Electronics PCB SERVICE

PROJECT	CODE	PRICE
JULY 2025		
Compact OLED Clock & Timer PCB	19101231	£4.99
180-230V DC Motor Speed Controller PCB	11104241	£9.99
Mains Power-Up Sequencer PCB (same as Feb '25)	10108231	£11.99
JUNE 2025		
Styloclone (case version)	23106241	£6.99
Styloclone (standalone version)	23106242	£7.99
Dual Mini LED Dice (TH LEDs, white solder mask)	08103241W	£2.99
Dual Mini LED Dice (TH LEDs, black solder mask)	08103241B	£2.99
Dual Mini LED Dice (SMD LEDs, white solder mask)	08103242W	£2.99
Dual Mini LED Dice (SMD LEDs, black solder mask)	08103242B	£2.99
Dual Hybrid Supply regulator board	18107211	£5.99
Dual Hybrid Supply front panel control board	18107212	£2.99
Dual Hybrid Supply CPU board	01106193	£3.99
Dual Hybrid Supply LCD adaptor	01106196	£2.99
MAY 2025		
Skill Tester 9000	08101241	£9.99
USB-C Serial Adaptor (black solder mask)	24106241	£2.99
DC Supply Protector (SMD adjustable version)	08106241	£3.99
DC Supply Protector (TH adjustable version)	08106242	£3.99
DC Supply Protector (TH fixed version)	08106243	£3.99
APRIL 2025		
ESR Test Tweezers PCB set	04105241+	£5.99
Fan Speed Controller Mk2	10104241	£4.99
Calibrated Microphone (SMD version)	01108231	£3.99
Calibrated Microphone (TH version)	01108232	£3.99
MARCH 2025		
Pico Gamer	08104241	£9.99
Compact Frequency Divider	04112231	£4.99
FEBRUARY 2025		
Laser Communicator transmitter	16102241	£3.99
Laser Communicator receiver	16102242	£3.99
Model Railway Points Controller transmitter	09101241	£3.99
Model Railway Points Controller receiver	09101242	£2.99
Mains Power-Up Sequencer	10108231	£11.99
JANUARY 2025		
USB to PS/2 Converter for VGA PicoMite	07111231	£3.99
ps2x2pico USB to PS/2 Converter	07111232	£3.99
Compact Microphone Preamplifier (case version)	01110231	£5.99
Compact Microphone Preamplifier (integrated version)	01110232	£5.99
DECEMBER 2024		
Raspberry Pi Clock Radio PCB pair	19101241/2	£12.99
VGA PicoMite (blue)	07107221	£5.99
Secure Remote Switch receiver	10109231	£4.99
Secure Remote Switch transmitter (module version)	10109232	£2.99
Secure Remote Switch transmitter (discrete version)	10109233	£2.99
NOVEMBER 2024		
Ideal Diode Bridge Rectifiers (28mm square)	18101241	£3.99
Ideal Diode Bridge Rectifiers (22mm square)	18101242	£3.99
Ideal Diode Bridge Rectifiers (4-pin SIL)	18101243	£3.99
Ideal Diode Bridge Rectifiers (tiny SOT-23)	18101244	£3.99
Ideal Diode Bridge Rectifiers (standalone SMD)	18101245	£4.99
Ideal Diode Bridge Rectifiers (standalone TH; 2oz copper)	18101246	£6.99
Multi-channel Volume Control (volume PCB)	01111221	£4.99
Multi-channel Volume Control (control PCB)	01111222	£4.99
Multi-channel Volume Control (OLED PCB, black)	01111223	£4.99

PROJECT	CODE	PRICE
OCTOBER 2024		
TQFP-32 Programming Adaptor	24108231	£4.99
TQFP-44 Programming Adaptor	24108232	£4.99
TQFP-48 Programming Adaptor	24108233	£4.99
TQFP-64 Programming Adaptor	24108234	£4.99
2m VHF FM Test Signal Generator	06107231	£4.99
SEPTEMBER 2024		
PIC Programming Adaptor	24105231	£4.99
Salad Bowl Speaker crossover	01109231	£7.99
AUGUST 2024		
Arduino ESR Meter combined L/C-ESR	04106182	£9.99
Arduino ESR Meter separate ESR	04106183	£7.99
Watering System Controller	15110231	£11.99
JULY 2024		
Dynamic NFC Tag (22 × 31mm)	06101231	£5.95
Dynamic NFC Tag (86 × 54mm)	06101232	£7.95
Reciprocal Frequency Counter	CSE230101C	£8.95
Reciprocal Frequency Counter front panel (black mask)	CSE230102	£7.95
Pi Pico-based Thermal Camera	04105231	£6.95
JUNE 2024		
Loudspeaker Test Jig	04106231	£12.95
MAY 2024		
GPS-Disciplined Oscillator	04103231	£12.95
Dual RF Amplifier	CSE220602A	£5.95
Songbird	08103231	£7.95
APRIL 2024		
Basic RF Signal Generator	CSE221001	£8.99
Basic RF Signal Generator front panel (black solder mask)	CSE220902B	£9.99
Silicon Chirp Cricket	08101231	£7.99
Test Bench 'Swiss Army Knife'	04110221	£9.99
MARCH 2024		
Digital Volume Control Potentiometer SMD version	01101231	£8.95
Digital Volume Control Potentiometer through-hole version	01101232	£9.95
Advanced SMD Test Tweezers body	04106221	£6.95
Advanced SMD Test Tweezers legs (pair)	04106212	£7.95
FEBRUARY 2024		
Active Mains Soft Starter	10110221	£10.95
JANUARY 2024		
Q Meter	CSE220701	£8.95
Q Meter front panel (black solder mask)	CSE220704	£7.95
Raspberry Pi Pico W BackPack	07101221	£8.95
DECEMBER 2023		
Digital Boost Regulator	24110224	£8.95
Dual-Channel Power Supply for Breadboards	04112221	£11.95
Display Adaptor for the Breadboard PSU	04112222	£9.95
NOVEMBER 2023		
LC Meter Mk3	CSE220503C	£9.95
DC Supply Filter for vehicles	08108221	£8.95
Discrete Audio Op Amp PCB	AO1-JUL23	£9.95
Discrete Audio Op Amp PCB + essential components	AO2-JUL23	£17.95

PCBs for most recent *PE/EPE* constructional projects are available. From the July 2013 issue onwards, PCBs with eight-digit codes have silk screen overlays and, where applicable, are double-sided, have plated-through holes, and solder mask. They are similar to photos in the project articles. Earlier PCBs are likely to be more basic and may not include silk screen overlay, be single-sided, lack plated-through holes and solder mask.

Always check price and availability in the latest issue or online. A large number of older boards are listed for ordering on our website.

In most cases we do not supply kits or components for our projects. For older projects it is important to check the availability of all components before purchasing PCBs.

Magazine back issues are available – see https://www.electronpublishing.com/product-category/pe-back-issues-print-and-digital/

Double-sided | plated-through holes | solder mask

PROJECT	CODE	PRICE
OCTOBER 2023		
Buck/Boost Charger Adaptor	14108221	£9.95
PIC Breakout Board for SOIC parts	24110225	£5.95
PIC Breakout Board for DIP parts	24110222	£5.95
AVR64DD32 Breakout board	24110223	£5.95
Automatic Train Controller	09109221	£6.95
Chuff Sound module	09109222	£6.95
SEPTEMBER 2023		
Mini LED Driver	16106221	£7.95
New GPS-Synchronised Clock	19109221	£9.95
AUGUST 2023		
110dB RF Attenuator	CSE211003	£9.95
Universal Battery Charge Controller (2023 update)	14107192	£9.95
Wide-Range OhmMeter	04109221	£12.95
JULY 2023		
Multimeter Checker/Calibrator	04107221	£10.95
JUNE 2023		
Arduino Programmable Load	04105221	£8.95
Buck-Boost LED Driver	16103221	£8.95
MAY 2023		
Precision AM-FM DDS Signal Generator	CSE211002	£14.95
Improved SMD Test Tweezers programmed PIC	0410621PIC	£12.95
AO 2x-dual-to-quad through-hole (pack of five)	AO1-MAY23	£6.95
AO 2x-dual-to-quad SMD (pack of five)	AO2-MAY23	£6.95
AO 2x-single-to-dual through-hole (pack of five)	AO3-MAY23	£6.95
AO 2x-single-to-dual SMD (pack of five)	AO4-MAY23	£6.95
APRIL 2023		
500W Amplifier Module	see p.22, April 2023	
Clipping Indicator (per channel)	01112211	£7.95
CD Welder Power Supply (one needed)	29103221	£9.95
CD Welder Controller (one needed)	29103222	£9.95
CD Welder Energy Storage module (several needed)	29103223	£7.95
AO Universal Dual Op Amp Board	AO1-APR23	£9.95
AO Stereo RIAA precision passives kit for dual op amp	AO2-APR23	£8.95
MARCH 2023		
Pico BackPack	07101221	£9.95
Semaphore Signal (controller)	09103221	£7.95
Semaphore Signal (blade)	09103222	£5.95
FEBRUARY 2023		
Tesla Coil driver board	26102221	£9.95
Tesla Coil potentiometer board	26102222	£5.95
Cooling Fan Controller & Loudspeaker Protector	01102221	£8.95
Remote Gate Controller	11009121	£12.95
JANUARY 2023		
Classic LED Metronome – 8-LED	23111211	£7.95
Classic LED Metronome – 10-LED	23111212	£8.95
Multi-Channel Speaker Protector – 6 channel	01101221	£9.95
Multi-Channel Speaker Protector – 4 channel	01101222	£7.95
Remote Control Range Extender – IR-to-UHF	15109212	£5.95
Remote Control Range Extender – UHF-to-IR	15109211	£7.95
AO Universal Single Op Amp Board	AO1-JAN23	£7.95
DECEMBER 2022		
Hummingbird Amplifier	01111211	£9.95
SMD Trainer PCB	29106211A	£8.95
SMD Trainer PCB + parts	29106211B	£13.95

PROJECT	CODE	PRICE
NOVEMBER 2022		
USB Cable tester – main PCB	04108211	£12.95
USB Cable tester – front panel	04108212	£5.95
USB Cable tester – optional panel	SC5970	£5.95
Model Railway Carriage Lights – PCB	09109211	£6.95
AO trnsfmr PCB – standard VTX-A range	VTX-101-007	£6.95
AO trnsfmr PCB – dual-outline VTX102-3001/101-3002	VTX-Dual	£6.95
OCTOBER 2022		
SMD Test Tweezers – PCB and pair of tweezer arms	04106211-2	£11.95
SMD Test Tweezers – programmed PIC12F1572-I/SN	0410621A	£7.95
Tele-com	12110211	£12.95
SEPTEMBER 2022		
Touchscreen Digital Preamp – main board	01103191	£12.95
Touchscreen Digital Preamp – adaptor board pair	01103192	£5.95
20A DC Motor Speed Controller	11006211	£9.95
AUGUST 2022		
Multi-purpose Battery Manager – I/O Expander	11104212	£5.95
Multi-purpose Battery Manager – Switch Module	11104211	£8.95
Simple MIDI Music Keyboard (for 8 switches)	23101213	£6.95
Nano Pong	08105212	£7.95

For the many earlier PCBs that we stock please see the PE website: www.electronpublishing.com

PE/EPE PCB SERVICE

Order Code	Project	Quantity	Price

Name ....

Address ....

....

Tel ....

Email ....

Card No ....

Valid From ....Expiry Date ....

Card Security No ....

You can also order PCBs by phone, email or via the shop on our website: www.electronpublishing.com

No need to cut your issue – a copy of this form is just as good!

All prices include VAT and UK p&p. Add £4 per *project* for post to Europe; £5 per *project* outside Europe.

Orders and payment should be sent to:

Practical Electronics, Electron Publishing Ltd
113 Lynwood Drive, Merley, Wimborne, Dorset BH21 1UU
Tel 07918 614662 Email: pe@pemag.au
Online Shop: www.electronpublishing.com

Sorry, cheques cannot be accepted as we are based in Australia and our UK bank account does not allow cheque deposits.

NOTE: Most boards are in stock and sent within seven days of order receipt. Please allow up to 28 days delivery if we need to restock.

Practical Electronics PCB SERVICE

PROJECT	CODE	PRICE
JULY 2022		
Silicon Labs AM/FM/SW Radio	CSE210301C	£10.95
Level Crossing Controller	09108211	£6.95
JUNE 2022		
Full-wave Motor Speed Controller	1010221	£8.95
PIC Programming Helper for 8-pin PICs only	24106211	£7.95
PIC Programming Helper for 8, 14 or 20-pin PICs	24106212	£10.95
Advanced GPS Computer	05102211	£9.95
MAY 2022		
Bus board PCB for Analogue Vocoder	AO1-MAY22	£10.95
Complete set of 14 PCBs for Analogue Vocoder	AO2-MAY22	£97.95
Programmed EEPROM for Digital FX Unit	FX1-MAY22	£10.95
Programmed PIC for Digital FX Unit w/ potentiometer	FX2-MAY22	£8.95
APRIL 2022		
64-key MIDI Matrix shield	23101211	£8.95
64-key MIDI Matrix switch board	23101212	£11.95
High-current Battery Balancer	14102211	£10.95
Digital FX Unit – using potentiometer	01102211	£9.95
Digital FX Unit – using BCD switch	01102212	£9.95
Universal Audio PSU	AO1-APR22	£11.95
MARCH 2022		
Mini Isolated Serial Link	24102211	£5.95
Busy Loo Indicator	16112201	£5.95
Analogue Vocoder – Band-pass filter board	AO1-MAR22	£9.95
Analogue Vocoder – HP/LP filter board	AO2-MAR22	£9.95
FEBRUARY 2022		
Arduino-based Power Supply	18106201	£9.95
Battery Monitor Logger	11106201	£10.95
Electronic Wind Chimes	23011201	£10.95
Analogue Vocoder – Driver Amplifier	AO-FEB22	£8.95
JANUARY 2022		
Vintage battery Radio Li-ion Power Supply	11111201	£9.95
MiniHeart: A Miniature Heartbeat Simulator	01109201	£8.95
DECEMBER 2021		
AM/FM/SW Digital Receiver	CSE200902A	£13.95
Balanced Input and Attenuator for USB CODEC	01106202	£11.95
NOVEMBER 2021		
Dual Battery Lifesaver	11111202	£6.95
OCTOBER 2021		
Mini Wi-Fi LCD BackPack	24106201	£8.95
SEPTEMBER 2021		
USB SuperCodec PCB	01106201	£14.95
Audio DDS Oscillator PCB	01110201	£5.95
Audio DDS Oscillator rotary encoder	01110201-ENC	6.95
Programming Adaptor Board for Audio DDS Oscillator	01110202	£5.95
High-power Ultrasonic Cleaner main PCB	04105201	£14.95
High-power Ultrasonic Cleaner front-panel PCB	04105202	
Night Keeper Lighthouse PCB	08110201	£5.95
AUGUST 2021		
Ol' Timer PCB	19104201	£11.95
Ol' Timer 8x8 RGB LED module using WS2812B	19104201-88	£8.95
Ol' Timer set of acrylic case pieces and spacer	19104201-ACR	£8.75
Ol' Timer DS3231 RTC IC wide SOIC-16	19104201-RTC	£5.95
Wideband Digital RF Power Meter	04106201	£9.75
Switchmode 78xx regulators ***(PACK of 5!)***	18105201	£7.95
Cool Beans SMAD display	CB-AUG21	£11.95

PROJECT	CODE	PRICE
JULY 2021		
ATtiny816 Breakout / Dev Board w/ Capacitive Touch	24110181	£9.75
IR Remote Control Assistant (Jaycar version)	15005201	£8.95
IR Remote Control Assistant (Altronics version)	15005202	£8.95
PIC18F Development Board	PNM-JUL21	£12.95
Microphone Preamplifier	AO-JUL21	£11.95
JUNE 2021		
Roadies' Test Signal Generator (surface-mount ver.)	01005201	£8.95
Roadies' Test Signal Generator (through-hole version)	01005202	£9.95
Touchscreen Wide-range RCL Box (Resistor module)	04104201	£18.95
Touchscreen Wide-range RCL Box (Ind/Cap module)	04104202	
KickStart Part 3 – Gyrator-based Audio Filter	KS3-2021	£7.95
MAY 2021		
7-Band Equaliser (Mono)	01104201	£8.95
7-Band Equaliser (Stereo)	01104202	£10.95
Car Altimeter	05105201	£7.95
APRIL 2021		
Reflow Oven – DSP Active Crossover (CPU)	01106193	£19.95
Reflow Oven – DSP Active Crossover (Front panel)	01106195	
Reflow Oven – DSP Active Crossover (LCD)	01106196	
Frequency Reference Signal Distributor	CSE200103	£8.95
MARCH 2021		
Nutube Guitar Effects Pedal	01102201	£12.95
Programmable Thermal Regulator (Peltier Interface)	21109181	£18.95
Programmable Thermal Regulator (Peltier Driver)	21109182	
Tunable HF Preamp	CSE190502	£8.95
FEBRUARY 2021		
4G Remote Monitoring	27111191	£9.95
JANUARY 2021		
Nutube Valve Preamplifier	01112191	£12.95
Arduino DCC Controller	09207181	£10.95
DECEMBER 2020		
Pseudo-Random Sequence Generator	16106191	£7.95
Clever Charger	14107191	£11.95
PE Theremin Amplifier	AO-1220-01	£8.95
NOVEMBER 2020		
LED Christmas Tree (1 off)	16107181-1	£6.95
LED Christmas Tree (4 off)	16107181-2	£14.95
LED Christmas Tree (12 off)	16107181-3	£24.95
LED Christmas Tree (20 off)	16107181-4	£34.95
USB/SPI Interface Board	16107182	£5.95
45V/8A Power Supply PCB *plus* acrylic spacer	18111181	£14.95
45V/8A Power Supply five-way display bezel	18111181-BZ	£3.95
Five-way LCD Panel Meter/Display	18111182	£7.95
OCTOBER 2020		
Digital Audio Millivoltmeter	04108191	£9.95
Precision Signal Amplifier	04107191	£7.95
SEPTEMBER 2020		
PE Theremin PSU	AO-0920-01	£5.95
PE Theremin PSU transformer	AO-0920-02	£7.95
Micromite Explore-28	07108191	£6.95
Ultrabrite LED Driver	16109191	£6.95
AUGUST 2020		
Micromite LCD BackPack V3	07106191	£9.95
Steering Wheel Audio Button to Infrared Adaptor	05105191	£7.95

Silicon Chip kits & parts for projects in this issue (posted from Australia):

Compact OLED Clock & Timer kit	https://siliconchip.au/Shop/20/6979	AUD $54 (about £27) + P&P
Hard-to-get parts for Dual Hybrid Power Supply regulator board	https://siliconchip.au/Shop/20/6096	AUD $120 (about £60) + P&P
Hard-to-get parts for Dual Hybrid Power Supply CPU board	https://siliconchip.au/Shop/20/6121	AUD $72 (about £36) + P&P

20% VAT is included in the above prices (it will be removed for customers outside the UK). Postage & packing cost is typically around £12-15. Please allow 1-2 weeks for delivery (longer in some cases).

CLASSIFIED ADVERTISING

Practical Electronics

If you want your advertisements to be seen by the largest readership at the most economical price then our classified page offers excellent value.

The rate for semi-display space is £10 per centimetre high, with a minimum height of 2·5cm. All semi-display advertisements have a width of 5.5cm. The prepaid rate for classified adverts is 40p per word (minimum 12 words).

For rates, further information on display and classified advertising or to book an ad, please email us at pe@pemag.au

Practical Electronics reaches more UK readers than any other UK monthly hobby electronics magazine. Our sales figures prove it.

We have been the leading monthly magazine in this market for more than thirty years.

BOWOOD ELECTRONICS LTD
Suppliers of Electronic Components
www.bowood-electronics.co.uk
Unit 10, Boythorpe Business Park, Dock Walk, Chesterfield,
Derbyshire S40 2QR. Sales: 01246 200 222
Send large letter stamp for Catalogue

Electrical Industries Charity (EIC)
We help people working in the electrical, electronics and energy community as well as their family members and retirees.
We use workplace programmes that give the industry access to financial grants and a comprehensive range of free and confidential services.
www.electricalcharity.org
electrical industries charity

COAST ELECTRONICS
BREAKOUTS-COMPONENTS-CONTRACT DESIGN-3D PRINTER PARTS-MUSICAL-MICROCONTROLLERS
WWW.COASTELECTRONICS.CO.UK

MISCELLANEOUS

VALVES AND ALLIED COMPONENTS?
For free stock list and/or advice, please contact me: geoffdavies337@gmail.com Telephone: 01788 574774

PIC DEVELOPMENT KITS, DTMF kits and modules, CTCSS Encoder and Decoder/Display kits.
Visit www.cstech.co.uk

COMPONENTS AS JOB LOT
Large stock. Transistors, Diodes, Logic, Memory and Analogue parts. 1000s of passive items. Catalogued and sorted. 'New Old Stock' in racks. Email: rojobarn@gmail.com

Andrew Kenny – Qualified Patent Agent
EPO UKIPO USPTO
Circuits Electric Machinery Mechatronics
Web: www.akennypatentm.com
Email: Enquiries@akennypatentm.com
Tel: 0789 606 9725

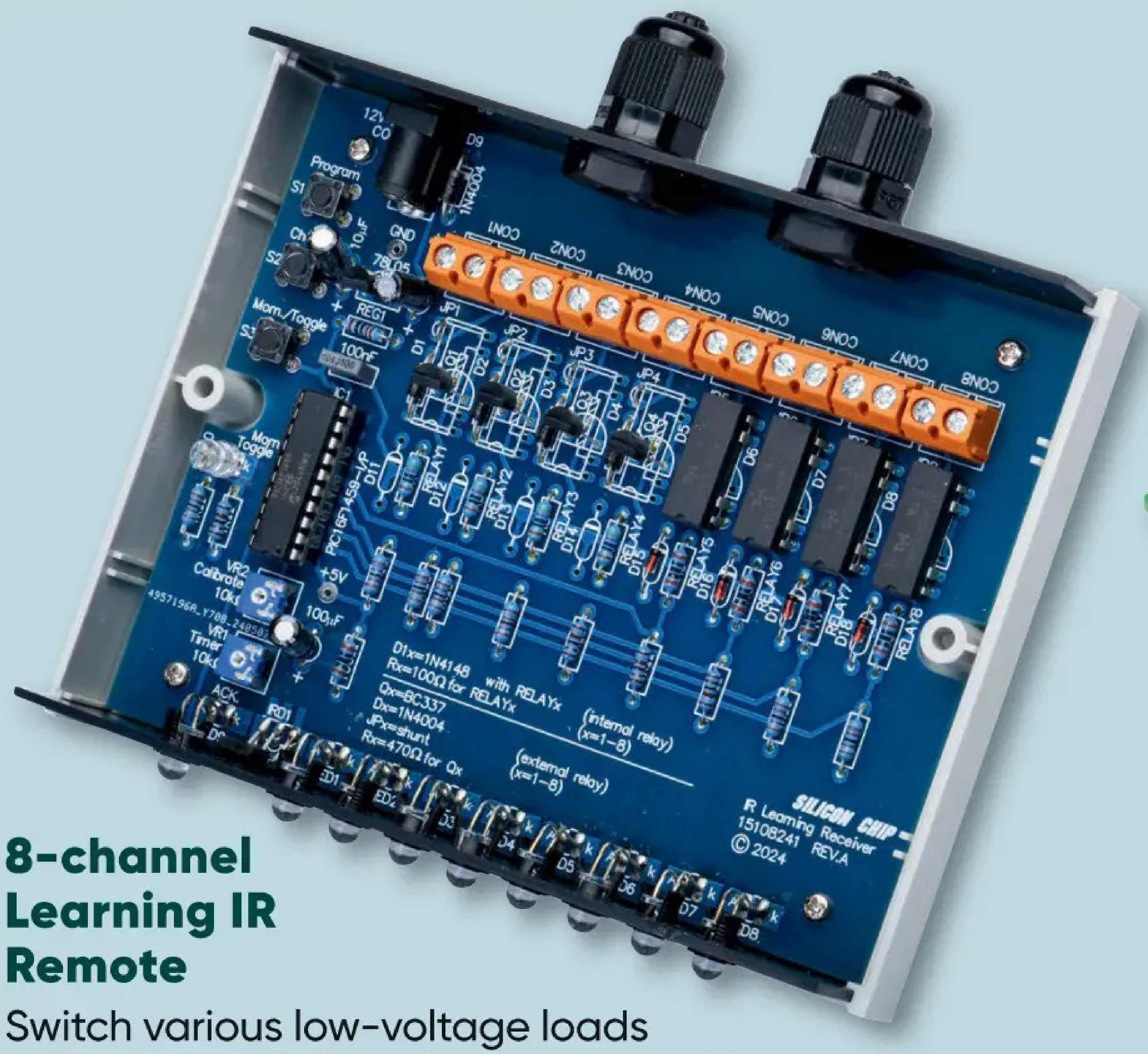
ADVERTISING INDEX

AOSHOP 56
BACK ISSUE PDFs. 33, 63
BASIC MICRO 25
ELECTRONICS TEACH-IN BUNDLES 51
ENIGMA COMPONENTS SHOP 13
HAMMOND MANUFACTURING. 62
MICROCHIP TECHNOLOGY . . . . . Outer back cover
PCBWAY. . . . . Inner front cover
SILICON CHIP MAGAZINE. 23
TAG-CONNECT 13
WIRELESS FOR THE WARRIOR BOOKS 45

Advertising mailing address
Electron Publishing Australia
PO Box 194
Matraville NSW 2036
Australia
Email pe@pemag.au
Web www.electronpublishing.com

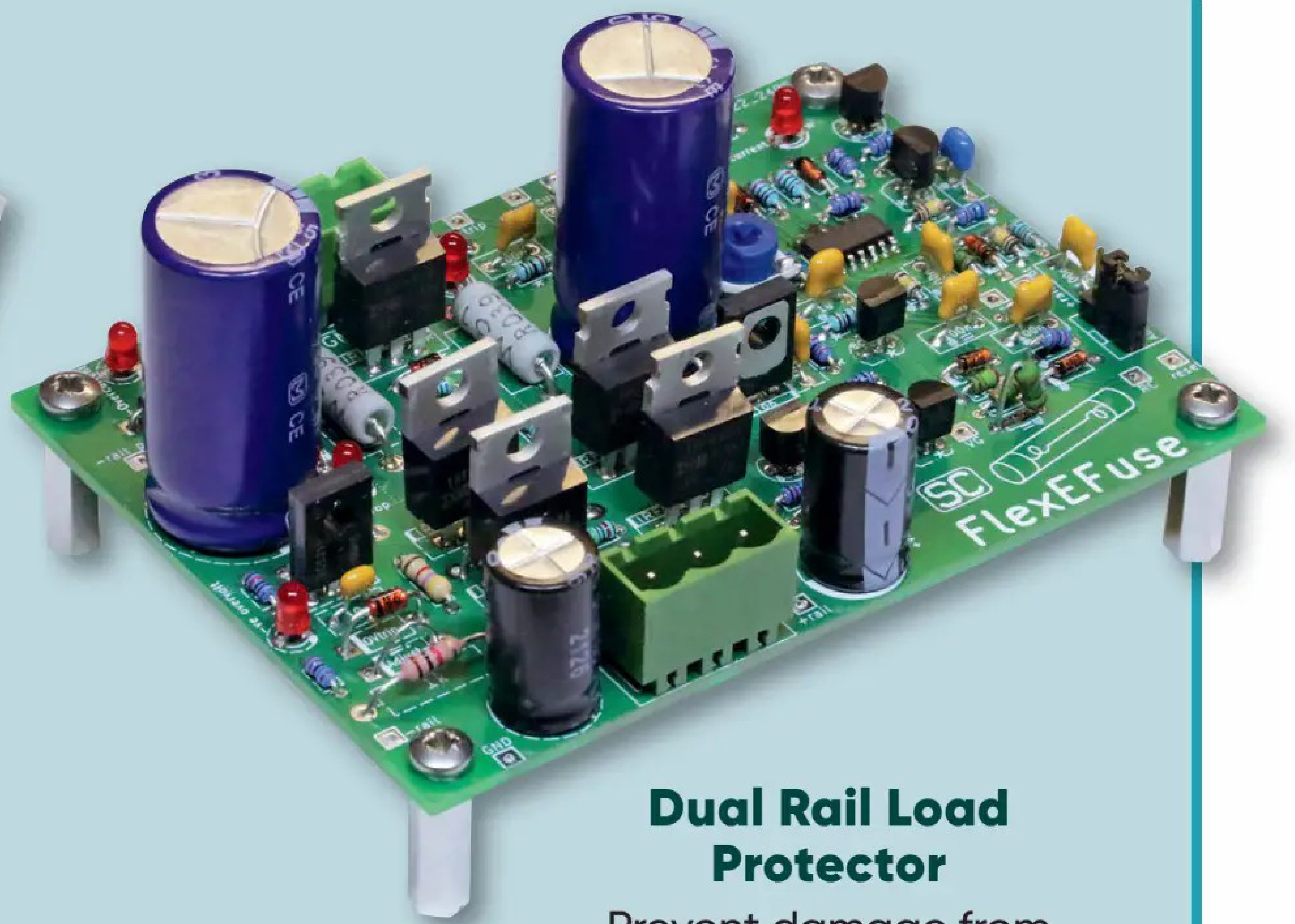
For editorial contact details, see page 2.

Next Month – in the August issue



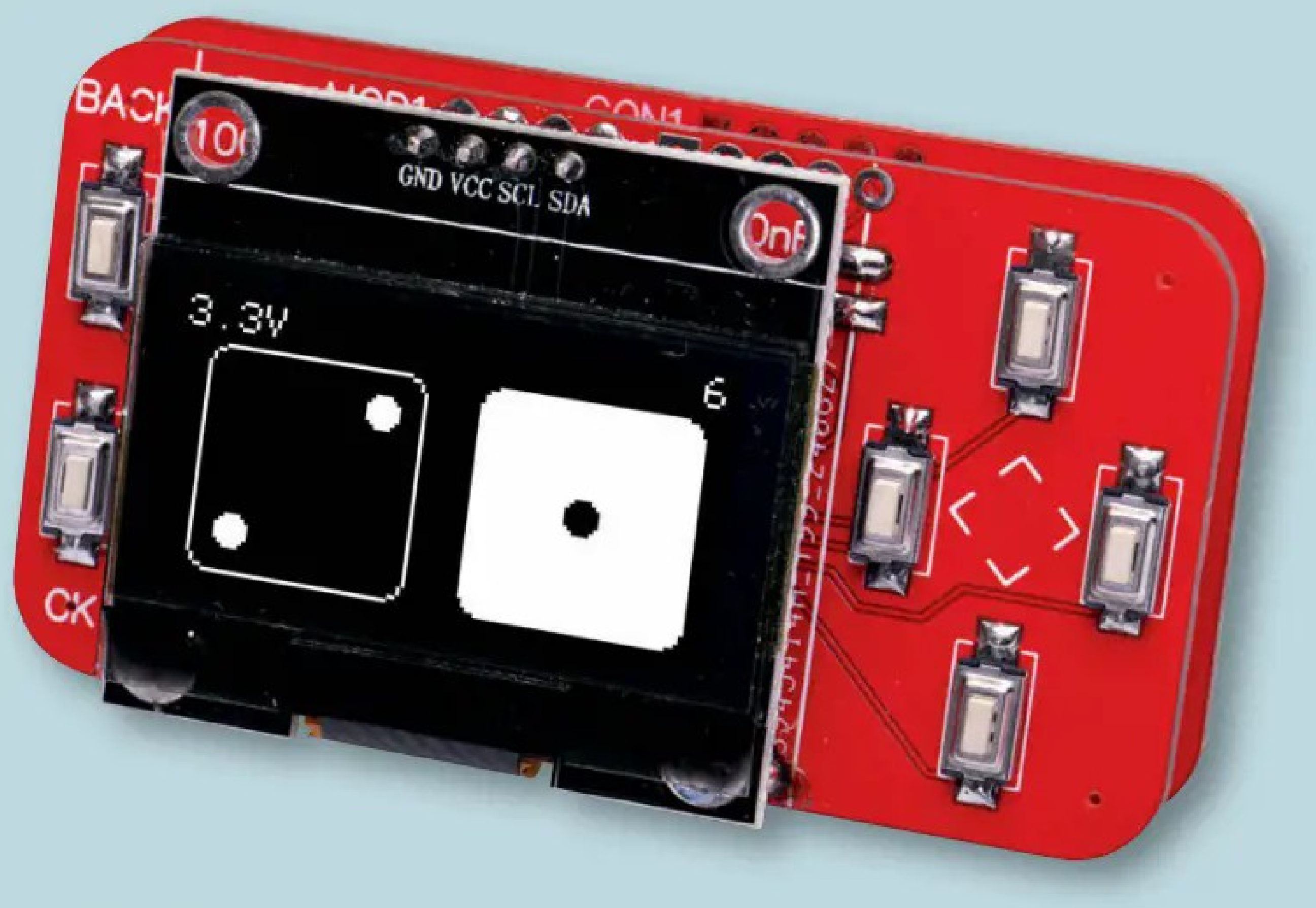
8-channel Learning IR Remote

Switch various low-voltage loads using a single infrared remote control



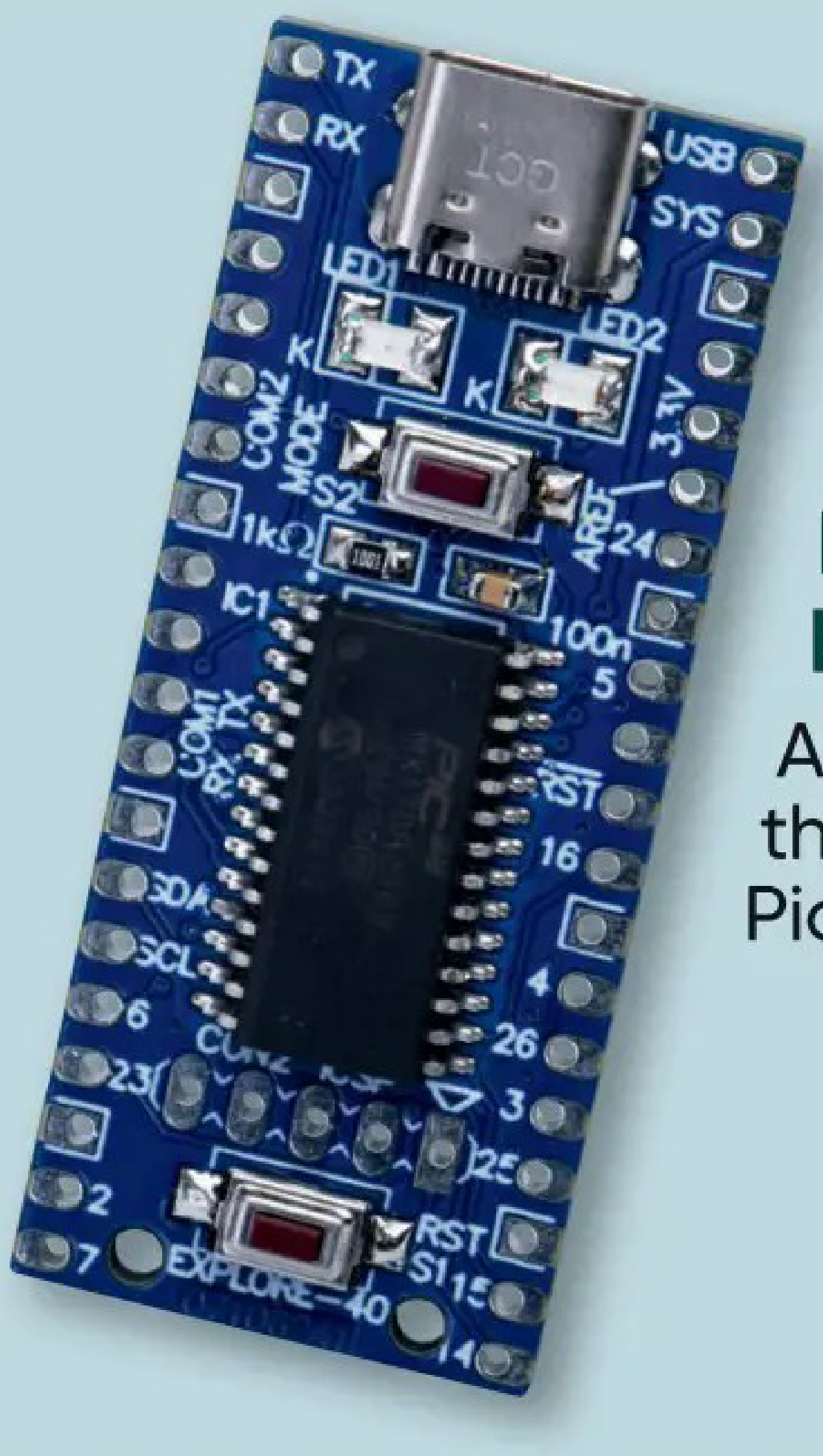
Dual Rail Load Protector

Prevent damage from incorrect supply voltages or excessive current



Flexidice

Can perform coin flips or roll one or two dice with up to 100 'sides'



Micromite Explore-40

A PIC board with the Raspberry Pi Pico form factor

On sale 3rd of July, 2025

Content may be subject to change

Practical Electronics

Subscriptions hotline

To subscribe or renew online, paying by credit/debit card or PayPal, please visit our shop at: www.electronpublishing.com

For those wishing to pay by direct debit, cheque or over the telephone, please call 'Select Publisher Services' on 01202 087631, or send a cheque (payable to: 'Select Publisher Services Ltd') with your details to:

Practical Electronics Subscriptions, PO Box 6337, Bournemouth BH1 9EH, United Kingdom

Remember, we print the date of the last issue of your current subscription in a box on the address sheet that comes with your copy.

Digital subscribers, please call 07918 614662 or visit: www.electronpublishing.com

Published on approximately the first Thursday of each month by Silicon Chip Publications, PO Box 194 Matraville NSW 2036 Australia. Printed in England by The Manson Group, Reynolds House, 8 Porters Wood, Valley Road Industrial Estate, St Albans AL3 6PZ. Distributed by Seymour, 86 Newman St, London W1T 3EX. Subscriptions UK: £33.99 (6mo); £59.99 (12mo.); £114.99 (2yrs). EUROPE: airmail service, £38.99 (6mo); £69.99 (12mo); £129.99 (2yrs). REST OF THE WORLD: airmail service, £46.99 (6mo); £84.99 (12mo); £164.99 (2yrs). Make payments to 'Practical Electronics', Practical Electronics Subscriptions, PO Box 6337, Bournemouth BH1 9EH United Kingdom. Email: pesubs@selectps.com. PRACTICAL ELECTRONICS is sold subject to the conditions that it shall not, without the written consent of the Publishers first having been given, be lent, resold, hired out or otherwise disposed of by way of Trade at more than the recommended selling price on the cover, and that it shall not be lent, resold, hired out or otherwise disposed of in a mutilated condition or in any unauthorised cover by way of Trade or affixed to or as part of any publication or advertising, literary or pictorial matter.

Did you know our online shop now sells the current issue of *PE* for £5.99 inc. p&p?

Practical Electronics
The UK's premier electronics, computing and maker magazine
Circuit Surgery
Setting up and timing a DAC for DSP.
Audio Out
An audio amplifier circuit using a complementary cascode
Precision Electronics
Getting the most out of your ADC in precision applications.
Compact OLED Clock & Timer
180-230V DC Motor Speed Controller with feedback
PLUS!
Techno Talk – humanoid robots are coming soon!
Cool Beans – generating predictably random numbers.
Staggering generator loads – using an existing design.
Jul 2025 £5.99
www.electronpublishing.com
@practicalelec
practicalelectronics
Precision Electronics
Bringing analog and digital together: ADCs & DACs
Dual Hybrid (switch-mode + linear) Bench Supply
Jun 2025 £5.99

You read that right! We now sell the current issue of your favourite electronics magazine for exactly the same price as in the High Street, but we deliver it straight to your door – and for UK addresses we pay the postage. No need to journey into town to queue outside the newsagent. Just go to our website, set up an account in 30 seconds, order your magazine and we'll do the rest.

www.electronpublishing.com

Cost-Effective PIC32A Microcontrollers With High-Speed Analog Peripherals

Supporting Complex General-Purpose Applications With Advanced Features

Enhance your embedded designs with the high-performance 32-bit PIC32A MCU family. Featuring a 200 MHz CPU, these MCUs excel in general-purpose and data-intensive applications. With an integrated double precision floating-point unit, they are ideal for math-intensive tasks. Designed for intelligent edge sensing and control, integrated high speed analog peripherals reduce system costs by minimizing external components. Built for safety-critical applications, they offer advanced safety and security features, making them suitable for connected medical, consumer, industrial and automotive applications.

Key Features

- 32-bit CPU with 200 MHz performance
- Single and double precision 64-bit Floating-Point Unit (FPU) coprocessor
- Two 40 Msps high-speed 12-Bit ADCs (25 ns conversion latency)
- Three 5 ns analog comparators
- Three 100 MHz operational amplifiers
- Secure Boot and Immutable Root of Trust
- Supports features that enable compliance with ISO 26262 and IEC 61508 Standards
- AEC-Q100 Grade 1 (125°C) qualification

Discover the full potential of the PIC32A MCU family. Visit our website for more information and to request product samples.

Autopilot
Voice Control

MICROCHIP
PIC32A MCUs
32-bit, High Performance

MICROCHIP

microchip.com/PIC32A

The Microchip name and logo and the Microchip logo are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries. All other trademarks are the property of their registered owners.
© 2025 Microchip Technology Inc. All rights reserved.
MEC2614A-UK-05-25